# Prediction of future input using binary codes can explain receptive field properties in primary visual cortex

Candidate 1030338


Word Count: 6,018

April 17th 2019

Presented for the degree of

MSc in Neuroscience

# Contents

# Acknowledgements

I would like to thank all members of the Auditory Neuroscience Group for their constant help and suggestions. In particular, Emil for introducing me to Tensorflow, Yosi for helping with convolution arithmetic and Nicol for directing my work. Also, this work would not have been possible without the help of Friedemann Zenke, who, in spite of my sheer stubbornness and frequent plodding, managed to instruct me on the fine art of machine learning.


# Author contribution

All the code written by the candidate in Python is available at a public repository (https://github.com/diegoasua/V1models.git). The Gabor Filter fitting algorithm, implemented in MATLAB, was taken from Singer et al. (2018). Throughout the dissertation the models are compared with published experimental data extracted from the original article (Jones and Palmer, 1987a), obtained from the correspondent author's personal website (Ringach, 2002) or provided directly by the authors (Niell and Stryker, 2008; Ohzawa et al., 1996)

# Abstract

Why do neurons in sensory systems respond the way that they do? Why do visual cortical neurons tend to respond best to oriented bars? One current hypothesis is that receptive fields of sensory neurons can be explained by the prediction of future input. In this study we have tested such hypothesis by developing a model with binary units tuned for visual motion prediction and comparing the properties of the artificial units to real neurons recorded in the primary visual cortex. We found that binary units develop receptive fields that resemble in many ways those of real neurons.

# 1 Introduction

Understanding what information of a stimulus is encoded in neuronal activity is a major goal in neuroscience. From the point of view of information theory (Shannon, 1948), one common approach is to assume that the human brain is optimised for efficiently encoding the information in a stimulus (Barlow, 2001). Another idea related to efficient coding is processing information through a sparse code, where the number of neurons carrying sensory information of a certain stimulus is minimised. Models optimising for sparse coding reproduce well the spatial tuning properties of neurons in primary visual cortex (Olshausen and Field, 1996, 1997; van Hateren and Ruderman, 1998). However, although there has been some success (Hateren and Schaaf, 1998; Carlson et al., 2012), it has been argued that sparse coding alone is not so capable of capturing the temporal tuning properties in primary visual cortex (Singer et al., 2018).

An alternative view of efficient coding is encoding relevant features of the stimulus and discarding those that are useless for guiding selection of appropriate motor responses (Bialek et al., 2001). One method to find features that may be useful for guiding action is finding those features that are predictive of the future; features in the world that tell us nothing about the future are unlikely to be relevant for guiding action. The concept of prediction as a driving force for perception stems back to Helmholtz (Helmholtz, 1924). A number of related frameworks are based on these ideas; 'predictive information', 'slow feature analysis', and 'temporal prediction'. These hypotheses with somewhat different assumptions posit that only features of sensory stimuli that are predictive of the near future are stored in neural activity (Sutton and Barto, 1981; Bialek et al., 2001; Palmer et al., 2015; Salisbury and Palmer, 2016; Heeger, 2017; Berkes and Wiskott, 2005; Berkes et al., 2009).

When evaluating putative normative models such as efficient coding, or temporal prediction or sparse coding it is a common practice to optimise the models for encoding natural stimuli and then to compare the tuning properties of the artificial units to experimentally

recorded receptive fields (RFs). Although several definitions have been proposed for RFs, a rather simple and clear one is that a RF is the linear transformation of the stimulus which evokes the maximum response (i.e. the highest firing rate) of the neuron (Adelson and Bergen, 1985; Reid et al., 1987). This linear RF forms the basis of another common phenomenological way to describe neural responses, the linear-nonlinear model, in which the output of a linear RF is then rectified or put through another kind of non-linear function (Ringach, 2004). With respect to the visual system, neurons in primary visual cortex (V1) show strong responses to sliding oriented bars (Hubel and Wiesel, 1959), and distinct spatio-temporal RFs.

Recently, a feedforward neural network optimised for temporal prediction — the efficient prediction of immediate future sensory inputs from recent past inputs — was shown to explain RF properties of neurons in primary auditory cortex, and of simple-cells in V1, most notably their temporal tuning properties (Singer et al., 2018). These findings have also been extended to a hierarchical temporal prediction model able to capture the different tuning properties along the visual pathway, from retina to higher visual cortex (Singer et al., 2019). However, the above temporal prediction models use units with non-negative real numbers as outputs, representing neural firing rates. An alternative view of neural responses is as a binary output — within a short time window neurons either spike or not.

This study will investigate the properties of a temporal prediction model with binary units and is divided into three specific aims:

1) To replicate the results from Singer et al. (2018) with a convolutional neural network.

2) To explore whether results obtained from the standard temporal prediction model can also be reproduced or extended by using hidden units with binary outputs.

3) To further explore the findings by introducing biophysically realistic integrate-and-fire units in the temporal prediction model.

# 2 Methods

## 2.1 Visual inputs

The data used for training the models consisted on a series of short movies of wildlife and landscapes with movement. Videos (sampled at 25 fps, without sound) were obtained from the same sources as described in Singer et al. (2018). Each frame was band-pass filtered to simulate a retinal filter (Olshausen and Field, 1997) and then downsampled by bilinear interpolation to obtain 100 x 160 pixel frames. Videos were then cut into sequential overlying clips, each of 6 frames duration (200 ms). The final training set was 39,600 clips, and the validation set was 4,400 clips. The first five frames of each clip was 'the past' and was the input to the network, and the final frame of each clip was 'the future' and was the target of the network. Finally, the training and validation sets were separately normalised by subtracting their mean and dividing by the standard deviation (over all pixels, frames and clips in the set). Ideally, the normalisation should be done only for the input ('the past') and only for the training set, and then the past for the validation set, and the future for the training and validation set have the mean of the past set subtracted and the results divided by the standard deviation of the past set.

## 2.2 The temporal prediction model

### 2.2.1 Implementation as a convolutional neural network

The temporal prediction model as described in Singer et al. (2018) starts by taking 20x20 patches around the centre of the frames across time steps in the clips. Then, the matrix containing the normalised pixel values is flattened and linearly mapped to a fully connected layer oh hidden units. Afterwards, a sigmoid non-linearity is applied and the output is then linearly mapped to a fully connected output layer, which yields a 20x20 patch of next time step of the clip. The prediction is then compared with the actual

future to calculate the error in the output, which is then backpropagated throughout the feedforward network. A scheme of the process is shown below (Figure 1).

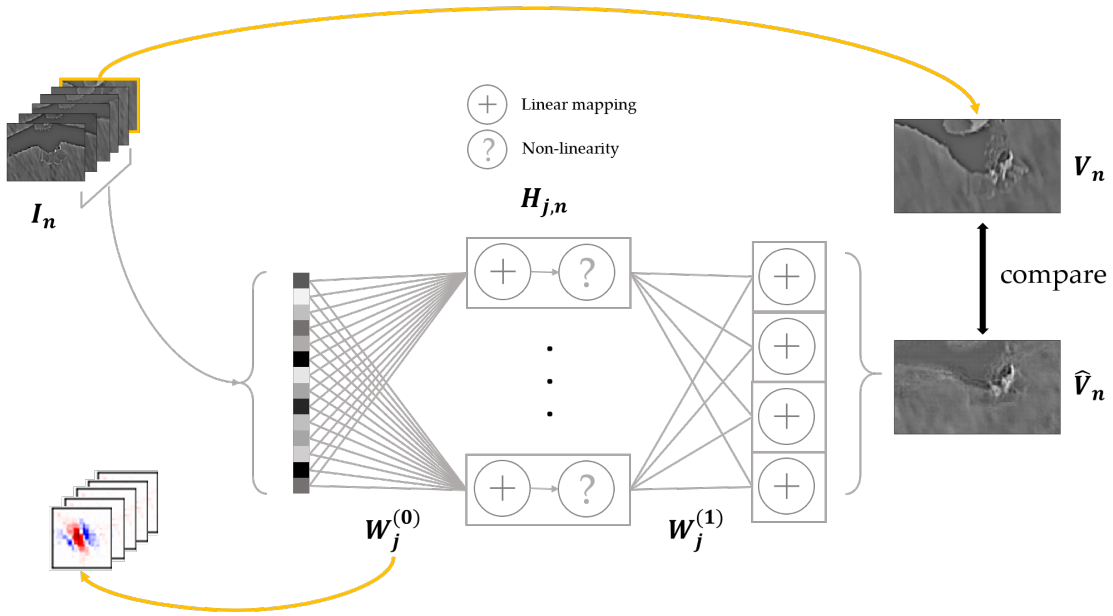


Figure 1: **The temporal prediction model with linear mapping in the input and output as reported in Singer et al. (2018)**. In this work we replaced the linear mapping by spatio-temporal convolutions. Basically, an input clip $\boldsymbol{U_n}$ consisting of five frames is linearly mapped by input weights $\boldsymbol{W_j^{(0)}}$ and then passed through a non-linearity, yielding the representation of the input in the hidden layer, $\boldsymbol{H_{j,n}}$. This is then linearly mapped to the output by output weights $\boldsymbol{W_j^{(1)}}$ and yielding the prediction $\boldsymbol{\hat{V}_n}$, which then is compared with the actual future $\boldsymbol{V_n}$ to generate the prediction error.

The model developed in this study is a modification of the original one. It consists of a spatio-temporal convolution of the past followed by a transposed convolution, yielding as an output one step forward into the future. In its most basic form, a convolution operation returns a filtered representation of the input by performing weighted sums of input patches with a sliding filter (also known as kernel). We denote the convolution operator by "$*$". In what follows, we will set down a mathematical formalisation of the model, in which any capital variable in bold stands for a rank 3 tensor.

For clip $n$, the input $\boldsymbol{I}_n$ is a tensor with two dimensions for space ($x = 1$ to $X$ and $y = 1$ to $Y$ elements) and one dimension for the temporal component ($t = 1$ to $T$ time steps). In particular, for the implementation of these models we set $X = 97$, $Y = 157$ and $T = 5$. $\boldsymbol{I}_n$ is then linearly convolved for each of the $j = 1$ to $J$ hidden units with input weights

$\boldsymbol{W}_j^{(0)}$ (i.e. kernels, with size $X^{(0)}$, $Y^{(0)}$ and $Z^{(0)}$), and then an input bias ($b_j^{(0)}$) is added for each unit. This convolution used a stride of 4 pixels over both spatial dimensions and 1 time-step over time; that is, the convolution is shifted in spans of 4 pixels over space, and spans of 1 time-step over time. This operation provides a linear mapping between the input and the hidden layer. In particular, in these models $X^{(0)} = Y^{(0)} = 21$ and $Z^{(0)} = 5$. Note , as the temporal span of the input $I$ is $T = 5$, and the temporal span of the kernel is $Z^{(0)} = 5$, the convolution over time in effect is just a dot product, however in the code it is written as a convolution and if $T$ was made larger a convolution over time is what would occur. The activity of unit $j$ for clip $n$ is $\boldsymbol{H}_{j,n}$, which results from applying a non-linearity $f$ to the output of the convolution:

$$\boldsymbol{H}_{j,n} = f\left(b_j^{(0)} + \boldsymbol{I}_n * \boldsymbol{W}_j^{(0)}\right) \tag{1}$$

The hidden units in the network have the same form as the well-established linear non-linear model. We created two different models according to different types of non-linear functions: A standard model that uses a sigmoid function and a binary model that uses a Heaviside function. The Heaviside function assigns 0 to any value below zero, and 1 to any value equal or greater than the threshold (in this case 0). A third, more complex model including spiking dynamics encoding the time dimension is described in detail in section 2.2.3. Regarding the transposed convolution, this can be viewed as a two step process. First, hidden representations are dilated with zero padding in-between elements ($\boldsymbol{H}_{j,n}^{dil}$) to account for the stride in the convolution. Second, the dilated representation of each unit $j$ is convolved with the corresponding output weights $\boldsymbol{W}_j^{(1)}$ (with size $X^{(1)}$, $Y^{(1)}$ and $Z^{(1)}$; in particular here $X^{(1)} = Y^{(1)} = 21$ and $Z^{(1)} = 1$). Finally an output bias $b_j^{(1)}$ is added. The output $\hat{\boldsymbol{V}}_n$ is the prediction for future of clip $n$:

$$\hat{\boldsymbol{V}}_n = b^{(1)} + \sum_{j=1}^{J} \boldsymbol{H}_{j,n}^{dil} * \boldsymbol{W}_j^{(1)} \tag{2}$$

Note that it is not possible to reconstruct the edges of the future frame as — in order to avoid artefacts — we did not add zero padding to the clip's spatial edges. Therefore, when comparing the actual future frame, $\boldsymbol{V}_n$, with the prediction, $\hat{\boldsymbol{V}}_n$ , to calculate the error in the output the future frame is evenly clipped to account for that.

### 2.2.2 Optimisation

The networks were implemented and trained using Tensorflow in Python (Abadi et al., 2016). Optimisation was performed with the efficient and robust gradient descent algorithm "Adam" with hyperparameters: $\hat{\epsilon} = 10^{-8},\ \beta_1 = 0.9,\ \beta_2 = 0.999$ (Kingma and Ba, 2014). $\hat{\epsilon}$ is the update parameter and $\beta_1$ and $\beta_2$ are the exponential decay rates for the moment estimates. The cost function minimised is given by the mean squared error of the output plus a regularisation term:

$$E = \frac{1}{NXY}\sum_{n=1}^{N}\left\|\hat{\boldsymbol{V}}_n - \boldsymbol{V}_n\right\|_2^2 + \lambda\left(\sum_{j=1}^{J}\left(\left\|\boldsymbol{W}_j^{(0)}\right\|_1 + \left\|\boldsymbol{W}_j^{(1)}\right\|_1\right)\right) \tag{3}$$

Where $\|\|_p$ is the p-norm of the tensor; p=1 is the sum of absolute values and p=2 is the square root of the sum of squared values, over all the values of the tensor. This prediction error in the output — given by the cost function — is backpropagated to both the weights and the biases ($\boldsymbol{W}_j^{(0)}$, $\boldsymbol{W}_j^{(1)}$, $b^{(0)}$ and $b^{(1)}$). In the second term of equation (3) we regularise both of the weight tensors using an L1 norm modulated by the regularisation strength $\lambda$. The regularisation helps to drive weights to zero unless they are useful in prediction, providing a parsimonious network. They can be interpreted as a constraint to minimise neural wiring. Although the gradient descent method is totally valid for training the standard model, training the binary model by this way resulted in a very poor improvement of the error even for the training set evaluation. This is due to the nature of the Heaviside function: It is discontinuous and binary; therefore, its derivative is null everywhere except at the firing threshold, where it is not defined. This makes it impossible to be optimised by classic gradient descent. One solution to this problem

is to evaluate the Heaviside function as-it-is during the forward propagation and then using a differentiable surrogate gradient during backpropagation. For the binary model we decided to implement a scaled sigmoid during backpropagation as a surrogate to the Heaviside, as it has been reported that training deep networks with binary functions by this way yields a robust training (Zenke and Ganguli, 2018).

The regularisation strength $\lambda$ and the scale of the surrogate gradient were chosen according to a criterion of best prediction error of the training set, given by equation (3) (see figures 2 and 3 on pages 16 and 17) after running the training algorithm for 50 epochs over the complete training set (for an explanation of why we chose the hyperparameters from the training set error and not from the validation set, see the discussion section 4.1). During training clips were split into mini-batches of 50 training examples for each epoch. Although this does not reach a minimum, the cost function does substantially flatten out, and this was the most we could do in the limited time. The models with the best hyperparameters, as determined by the method above, were further run for 200 epochs. This again did not entirely reach the minimum, and time was limited. The model would have been run to a definite minimum for all hyperparameter settings given more time.

### 2.2.3 Spiking version

A further extension of the model consists in adding spiking temporal dynamics. The aim of this development was to further increase the realism of the units in the model. Due to the complexity of this development, for simplicity a convolutional model was not used, instead the basic non-convolutional model reported in Singer et al. (2018) was modified to spiking units with biophysically-realistic dynamics. Therefore, for this model the dataset was processed differently. Each filtered video was divided up into 21 pixel x 21 pixel x 50 time-step spatio-temporal patches. These constitute the clips. The clips were divided into training and validation sets, and normalized as for the other models. Each frame of clip $n$, from $t = 1$ to $T$ (in particular, here $T = 50$) was input to the network one by one in an iterative manner. This way, the temporal component is encoded in the recurrent

dynamics of the spiking units in the hidden layer. At each time step $t$ the input patch of the clip is linearly mapped by the input weight to each of the units in the hidden layer, resembling what would be the input current $I$ to each of the $j = 1$ to $J$ neurons. These units are modelled as Izhikevich's quadratic integrate-and-fire neurons (Izhikevich, 2003), defined by a system of two ordinary differential equations:

$$\frac{dv}{dt} = 0.04v^2 + 5v + 140 + I - u \tag{4}$$

$$\frac{du}{dt} = a\,(bv - u) \tag{5}$$

Here, $v$ is the membrane potential in mV and $u$ the membrane recovery variable, which provides negative feedback to $v$. Parameters $a$ and $b$ are dimensionless and can be varied to model different classes of neurons and $t$ the time in ms. In addition, the model also depends on an auxiliary function that resets v and u after a spike:

$$\text{if } v \geq 30 \text{ mV, then } \begin{cases} v \leftarrow c \\ u \leftarrow u + d \end{cases} \tag{6}$$

In this equation the parameter $c$ is the reset value of the membrane potential — in mV — after a spike, and $d$ is another dimensionless parameter that modulates the reset membrane recovery variable. Here we have set the parameters of the quadratic integrate-and-fire model to those values that resemble those of regular spiking cortical neurons ($a = 0.02$, $b = 0.2$, $c = -65$ mV, $d = 8$) as described in Izhikevich's paper (Izhikevich, 2003). In the actual code, discretized versions of equations (4) - (6) were used, discretized by Euler's method. The output of the neuron at each time step is whether the neuron has spiked or not. Each output is then linearly mapped by output weights to generate a prediction of the frame at time $t + 1$. After each time step the prediction error of the

output — equivalent to equation (3) — is generated by comparing the $t + 1$ prediction with the actual $t + 1$ patch. Equation (6) is indeed a discontinuous function, and as the Heaviside function, it needs to make use of a surrogate gradient in order to allow optimisation of the input weights and biases. Following that reasoning, the network is optimised as described in section 2.2.2 for the Heaviside non-linearity.

## 2.3 Calculation of RF power in time

The power of an RF in time step $t$ is defined as the squared values of the RF averaged over space. The power of a set of RFs over time, is the power over time of each unit, averaged over all units ($j = 1$ to $J$. The fraction of RF power in time is an estimate of the relevance of each time step into the past for predicting the near future. The power of a set of RFs in time step $t$ is defined as the squared values of such RFs averaged over space and over all units ($j = 1$ to $J$).

## 2.4 Estimating space-time separability

One property of the generated receptive fields is time separability. If the space-time structure of a RF can be decomposed into a product of a temporal function and a space function then the RF is deemed separable. This definition allows us using singular value decomposition across time to discriminate which of the units were space-time separable and which were inseparable. To be comparable with previous experimental studies, the latter units were not included in quantitative analyses involving Gabor fitting that are outlined below (Ringach, 2002; Singer et al., 2018). As described in Singer et al. (2018), a unit $j$ is space-time separable if its first singular value is at least two times greater than its second singular value. For any other case the unit was considered inseparable, and therefore was further processed.

## 2.5 Fitting Gabor filters to receptive fields

To measure different properties of RFs to quantitatively compare the ones from the models with experimental ones, one common way is to fit a Gabor function to the RF (Jones and Palmer, 1987a). We fitted Gabor functions to each RF at the time step at which its power was greatest (see section 2.3) using the fitting algorithm described in Singer et al. (2018) (including the anti-aliasing correction). The Gabor function is defined as follows:

$$G\left(x^{'}, y^{'}\right) = A\, exp\left(-\left(\frac{x^{'}}{\sqrt{2}\sigma_x}\right)^2 - \left(\frac{y^{'}}{\sqrt{2}\sigma_y}\right)^2\right)\cos\left(2\pi f x^{'} + \phi\right) \tag{7}$$

$(x^{'}, y^{'})$ is the set of spatial coordinates of the Gabor filter that arise from translating the centre of the RF $(x_0, y_0)$ to the origin and rotating it $\theta$ rad:

$$x^{'} = (x - x_0)\cos(\theta) + (y - y_0)\sin(\theta) \tag{8}$$

$$y^{'} = -(x - x_0)\sin(\theta) + (y - y_0)\cos(\theta) \tag{9}$$

The parameters $\sigma_x$ and $\sigma_y$ describe the width of the Gaussian envelope in $(x^{'}, y^{'})$ directions. $\theta$ is the spatial orientation, and the parameters $f$ and $\phi$ correspond to the spatial frequency and phase only along the $x^{'}$ direction. Finally, A describes the height of the Gaussian envelope.

The fitting algorithm is complex and its implementation is carefully detailed in Singer et al. (2018). For a RF to be fit it must fulfil a set of 3 criteria: Pixel wise correlation between fitted Gabor and real RF $\geq 0.7$; $(x^{'}, y^{'})$ must fall inside the space of the RF; and both $\sigma_x$ and $\sigma_y$ must be $> 0.5$. The last criterion of the three is because many of the fit parameters are hard to determine accurately when the Gaussian envelope is very small, near the size of a pixel.

## 2.6 Analysis of RF structure

In order to quantitatively compare the structure of the fitted Gabor filters we used two previously defined measures (Jones and Palmer, 1987a): $n_y = \sigma_y f$ provides a measure of how long the bars of the filter are and $n_x = \sigma_x f$ is an indirect measure of the number of alternating bars in the Gabor filter. Only those units that were space-time separable and fulfilled the Gabor fitting criteria (see section 2.5) were included in this analysis. The two dimensional scatter provided by $n_x$ and $n_y$ gives an idea of the different shapes present in a population of Gabor filters, allowing quantitative comparison between sets of data. In addition, temporal and spatial frequencies were calculated as their respective peaks from the 2D Fourier transform of the spatio-temporal structure of the receptive fields.

# 3 Results

## 3.1 Hyperparameter tuning

We built a neural network whose goal was efficiently predict the final frame of each clip — the immediate future — from the first five frames — the past. The model is a convolutional version of the the temporal prediction model developed in Singer et al. (2018), where the authors used a single-hidden-layer feedforward neural network with L1 weight regularisation to predict the immediate future frames of natural movie clips from past frames. This model involved a simple linear mapping followed by a sigmoid non-linearity to provide the output of the hidden layer units and from there a simple linear mapping from the hidden unit output to predict the immediate future frame (Figure 1). Because of the linear nature of convolutions we expected that a convolutional version of the temporal prediction model (standard model) would largely replicate their findings. In addition, we made a second version of the model by replacing the sigmoid non-linearity by a Heaviside function, this way converting the output into a binary outcome (binary model) as a first approach to a spiking temporal prediction model. First, we trained the networks for 50 epochs for a range of values of the hyperparameters: we varied the regularisation strength of both models, and also the scale of the surrogate-gradient of the binary model. We then chose the hyperparameter settings of the models that gave the lowest prediction error (Figures 2, 3, S1 and S2). Because of the uneven form of validation curves (Figures 2 and 3), perhaps due to some limitations of our training process and dataset (see Discussion), we were not confident in the minimums that they showed, so we instead picked the optimum hyperparameters using the predictions on the training set. With more time to train our models and check our validation set, we would instead have picked our hyperparameters using the validation set, which we recognise is the more standard method. We show plots of prediction error for both the training and validation sets — it can be seen that the minimums are not radically different, especially for the binary model. We found that the best regularisation strength was $10^{-5.75}$ for the model

trained with a sigmoid non-linearity (Figure 2).

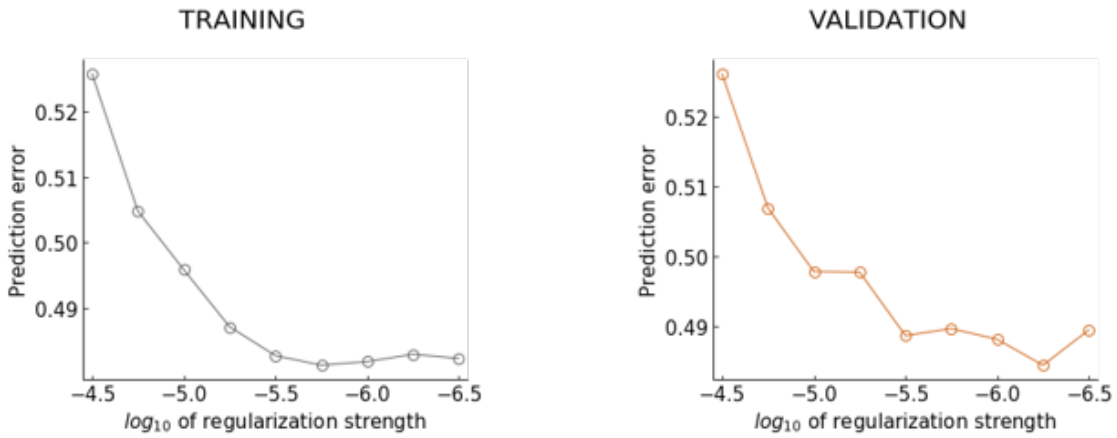


Figure 2: **Tuning regularisation strength of the standard model.** Prediction error in the output measured as the mean squared error (left: training set; right: validation set) after 50 training epochs as a function of the regularisation strength (logarithmic units).

For the binary model the optimal scale of the surrogate gradient was 100 (Figure 3A). (see details of how the binary model is trained in section 2.2.2) and the optimum regularisation strength was $10^{-5}$ (Figure 3B). These are the values we proceeded with in the following analysis.

In both cases the hidden layer was made up of 128 linear non-linear units in the hidden layer. Doubling or triplicating the number of units resulted in little improvement of the prediction error with an associated heavy increase of the computation cost. Also, by minimising the number of units in the hidden layer we facilitate generalisation and prevent overfitting. The learning rate used during the training process was 5 x $10^{-4}$, this being the highest at which prediction error of the next frame was not compromised, and therefore, yielding the quickest training at the lowest possible prediction error. After tuning hyperparameters to produce the best prediction error we further ran both models with their optimal parameters for 200 epochs to further optimise the network toward a steady state (S3).

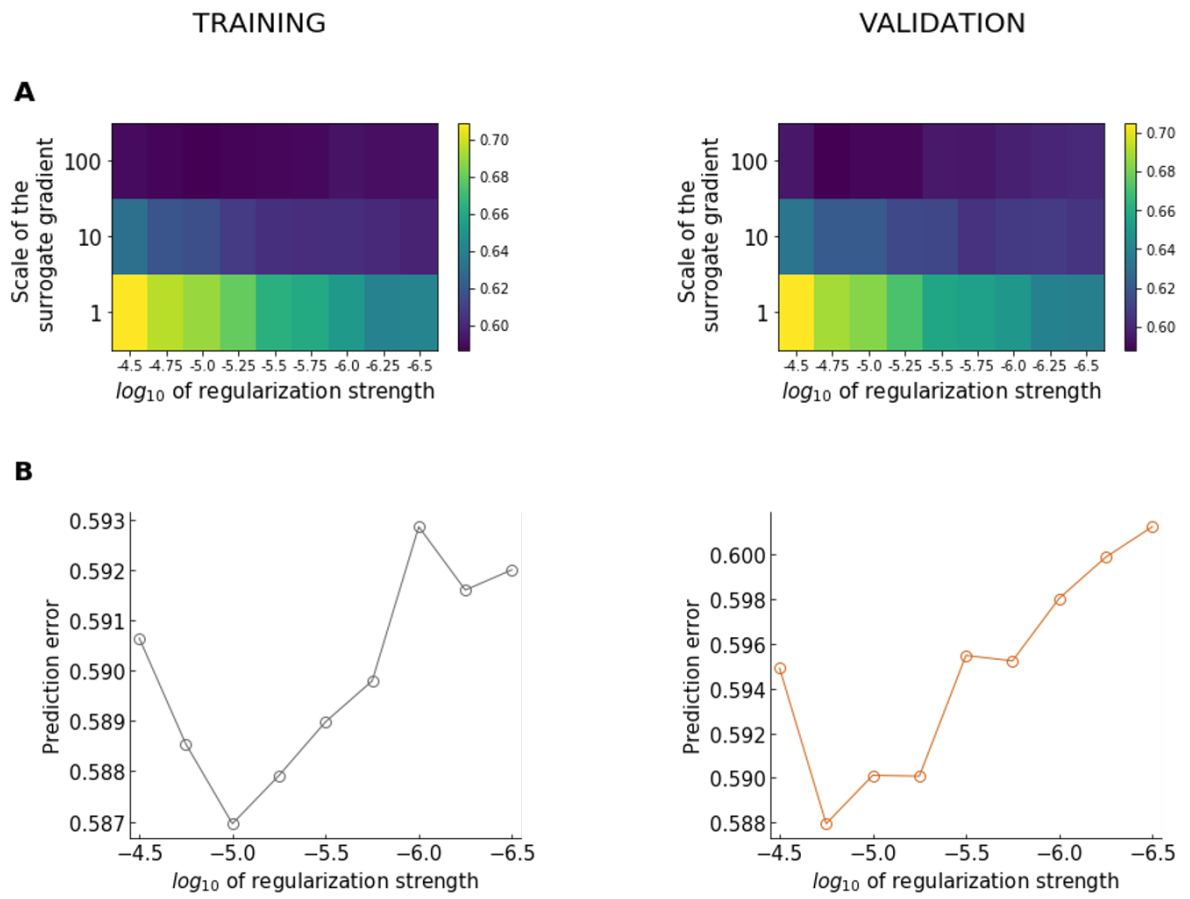


Figure 3: **Hyperparameter search for the binary model.** (A) Heatmat representing prediction error in the output measured as the mean squared error (left: training set; right: validation set) after 50 training epochs as a function of the regularisation strength (logarithmic units) and the scale of the surrogate gradient. Colour represents prediction error, with scale bar on the right. (B) Prediction error for a fixed scale of the surrogate gradient of 100 (best prediction).

## 3.2 RFs developed by the models

Next we extracted the input weights from the networks and compared them with experimentally acquired RFs in V1. Figure 6 shows some full spatio-temporal RFs from real V1 (Figure 6A-I), the standard model (Figure 6A-II), and the binary model (Figure 6A-III). Observe how for the data and the model the spatio-temporal RFs' power decays into the past from the present (0 ms).

Figures 4-5 plot the spatial RFs of all the units of the standard and binary models respectively. Each spatial RF is at the time-step with the highest power (sum over space of squared values) of the spatio-temporal RFs, which is generally at the present (0 ms)

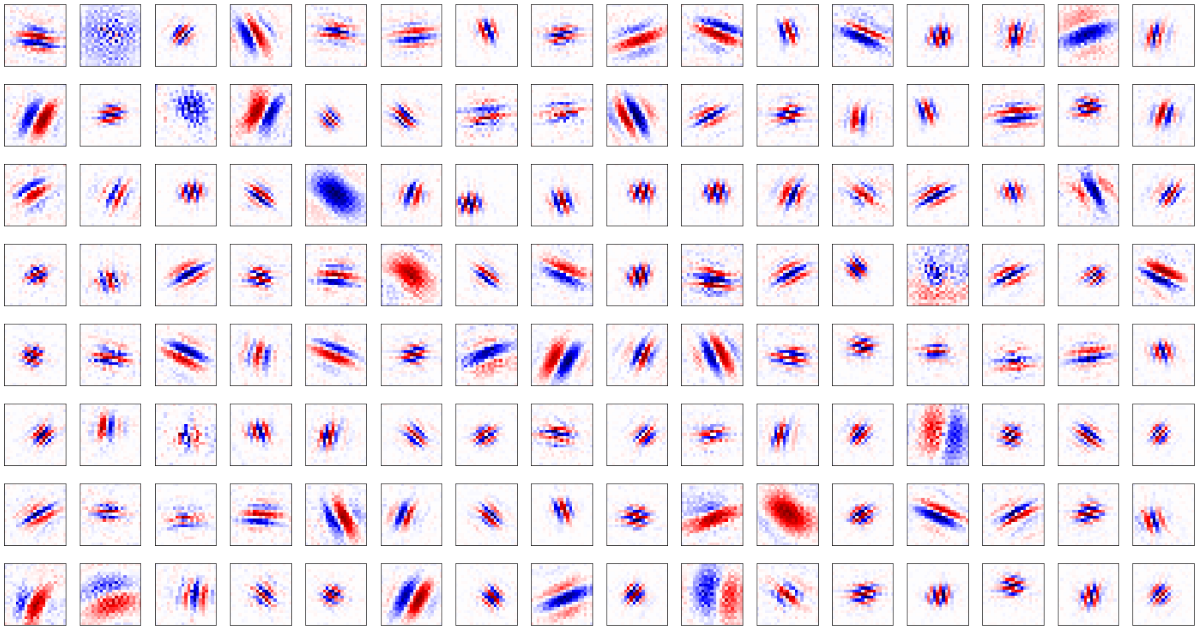

Figure 4: **Complete set of receptive fields generated by the standard model at their highest power.** Colour represents 'on' and 'off' regions.

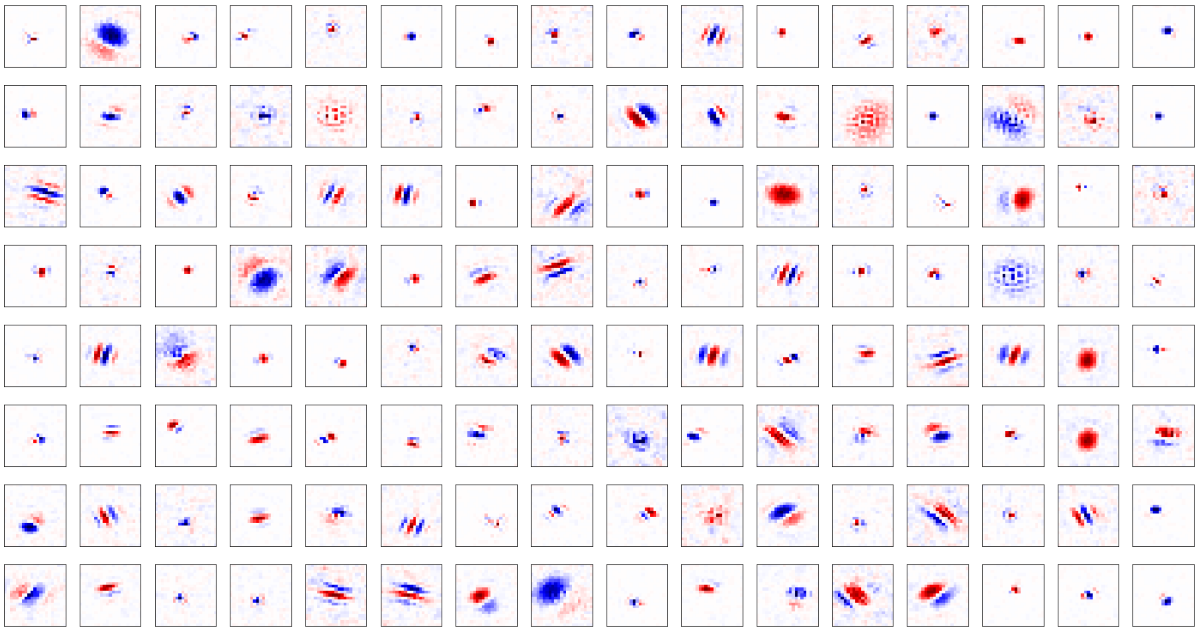

Figure 5: **Complete set of receptive fields generated by the binary model at their highest power.** Colour represents 'on' and 'off' regions.

or preceding (-40 ms) time-step. The RFs produced by both models are diverse and capture many different features of real data — notably Gabor filter-like structure is often seen (Figures 4-5). Interestingly, the binary model produces a richer variety of filters than the standard temporal prediction model, including centre-surrounded RFs that are found for a minority of units in V1 (Zylberberg et al., 2011). Figures S4-S11 show the

RFs for different regularisation strengths of the model (same model runs as in Figures 2-3). These figures indicate that the difference described above between the binary and standard model appears to be true regardless of the strength of regularisation, with centre-surround RFs being present in the binary model at a range of strengths, but not in the standard model over a similarly wide range of regularisation strengths. Also, RFs in the binary model were in general smaller than those generated by the standard model (Figures 4-5). This also appears to hold true over a range of regularisation strengths (Figures S4-S11), except perhaps for the binary model at the weakest regularisation strengths where the RFs appear approximately comparable in size to those of the standard model at the highest regularisation strengths.

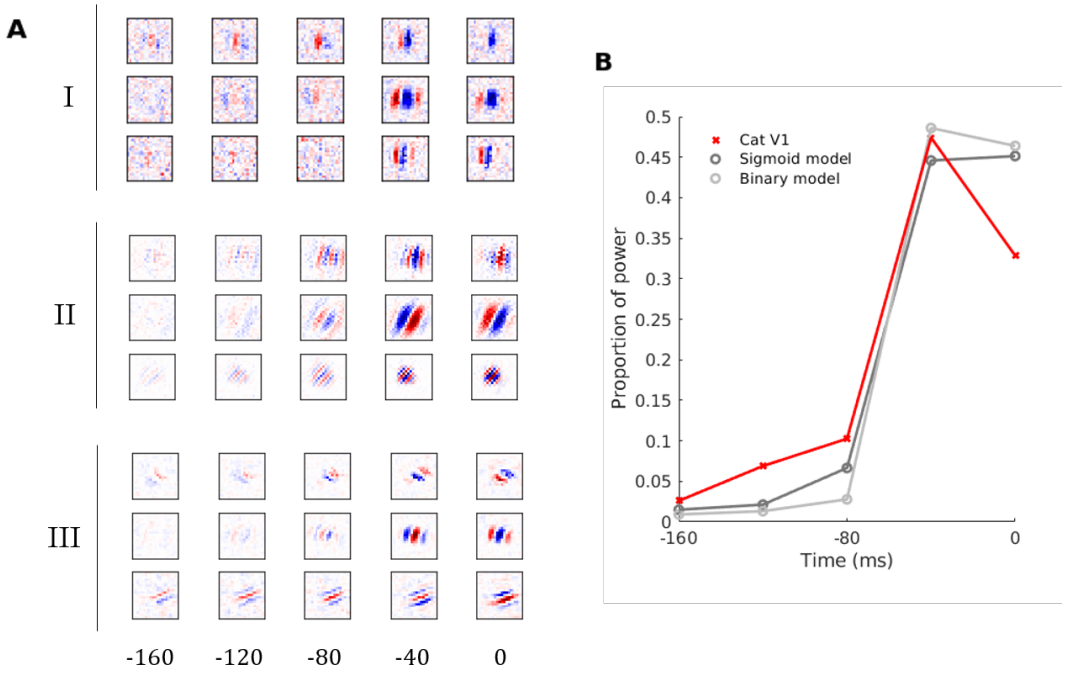


Figure 6: **Model units capture the temporal asymmetry of experimental RFs**. (A) Representative spatio-temporal RFs recorded in cat V1 from Ohzawa et al. (1996) (I), developed by the standard model (II) and developed by the binary model (III). (B) Quantification of RF power over time. Time scale in (A) is as in (B).

After this visual inspection, we performed quantitative analyses on the binary and standard model that was run for 200 epochs. Ideally this analysis would be done at all the regularisation strength, but there was insufficient time to do this.

## 3.3 Temporal properties of the modelled RFs

We first compared the temporal properties of the generated RFs with spatio-temporal RFs measured in cat V1 (Ohzawa et al., 1996). We found that RF power is highest nearest the present and decays into the past and decays into the past resembling experimental data (Figure 6). Both models use almost exclusively information contained during the previous 80 ms, congruent with the fact that experimental data shows accumulation of more than 95% of the power in that same interval.

## 3.4 Spatial tuning properties of modelled RFs

In order to compare RF tuning properties we fitted Gabor filters to both experimental and modelled RFs at the time step with the highest power. Then we measured the orientation and spatial frequency tuning of each of the units. The orientation tuning being orthogonal to orientation of the bars of the fitted Gabor, and the spatial frequency tuning being the spatial frequency of the sine wave component of the fitted Gabor.

For the standard temporal prediction model 42.2% (54/128) of units were space-time inseparable and fulfilled the fitting criteria and were therefore analysed; for the binary model this value dropped to 35.2% (45/128). This may be due to the fact that the binary model develops a subpopulation of RFs that are not Gabor filter-like. The fitted Gabor filters spanned over a wide range of orientation and spatial frequency tuning (Figures 7A, 8A). Interestingly, whereas the binary model showed a preference for spatial orientations spanning along vertical and horizontal axes, the standard model orientation distribution did not. Results from mouse V1 suggest a preference for vertical and horizontal tuning in neurons (Kreile et al., 2011). One detail to note is that the standard model developed RFs which appear somewhat clustered (Figure 7A, top).

We also produced an $n_x$-$n_y$ plot (see Methods) of the RFs to examine RF shapes and compare them to the data; units with high $n_y$ values have long bars in their RFs, units

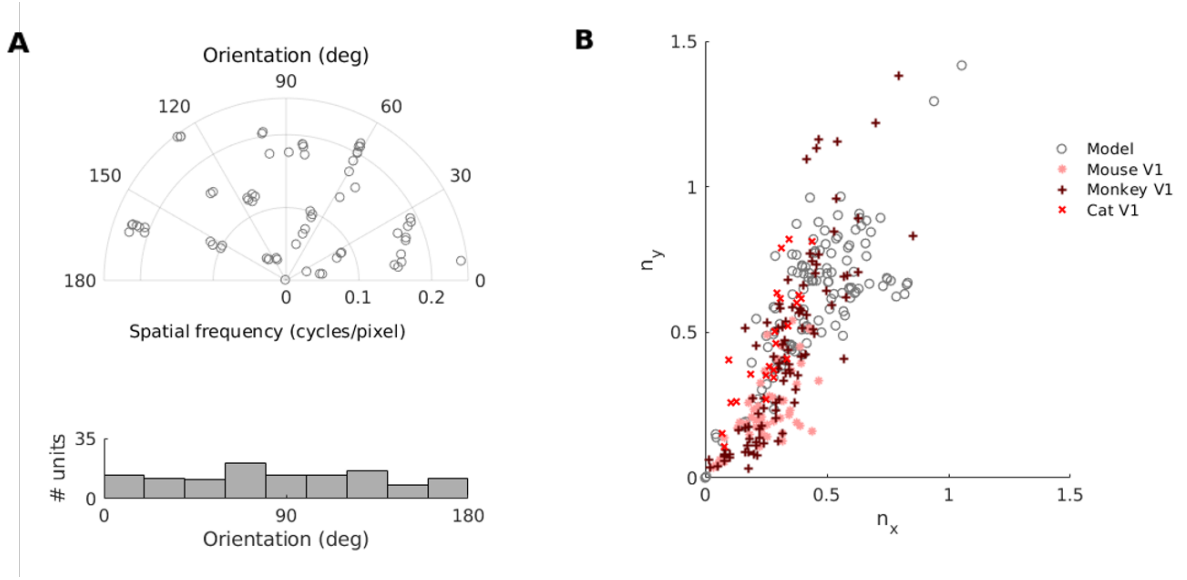


Figure 7: **Spatial properties of the standard model's RFs.** (A) Orientation and spatial frequency distribution of the modelled RFs. (B) Distribution of RF shapes for real neurons (mouse (Niell and Stryker, 2008), monkey (Ringach, 2002), cat, (Jones and Palmer, 1987b)) and standard model units. $n_x$ and $n_y$ measure RF span parallel and orthogonal to orientation tuning, as a proportion of spatial oscillation period (Ringach, 2002).

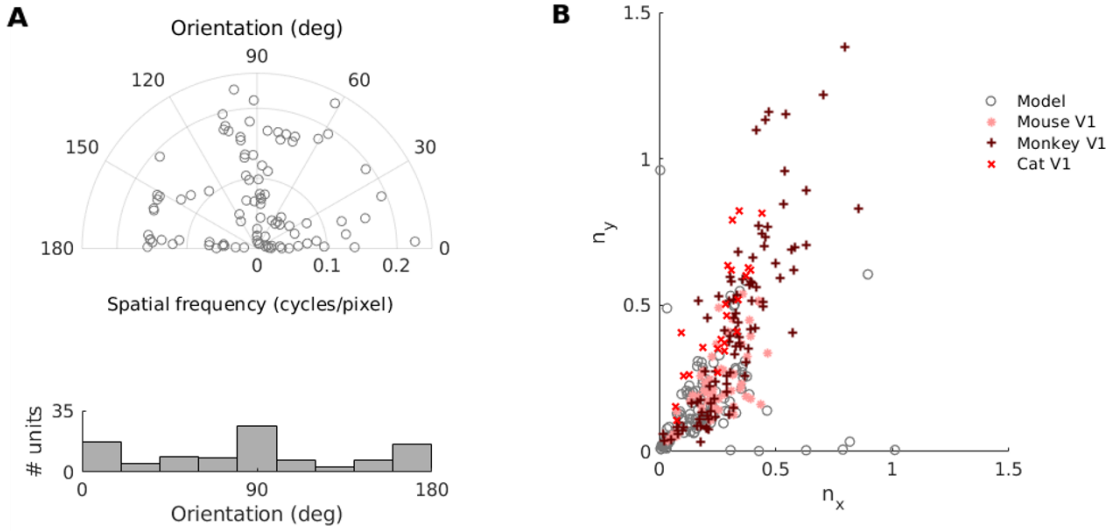


Figure 8: **Spatial properties of the binary model's RFs** (A) Orientation and spatial frequency distribution of the modelled RFs. (B) Distribution of RF shapes for real neurons (mouse (Niell and Stryker, 2008), monkey (Ringach, 2002), cat, (Jones and Palmer, 1987b) and binary model units. $n_x$ and $n_y$ measure RF span parallel and orthogonal to orientation tuning, as a proportion of spatial oscillation period (Ringach, 2002).

with high $n_x$ values have many bars in the RFs (see Methods). Blob-like RFs are found near the origin, whereas stretched RFs with many on/off regions have high $n_x$ and $n_y$

values (Ringach, 2004). Interestingly, the widths and lengths of the RFs of the binary model relative to their oscillation phase were consistent with those from neural data from mouse, monkey and cat V1 neurons, although they failed to account for a subpopulation of monkey V1 RFs with large $n_x$ and $n_y$ values (Figure 8B). In contrast, the standard model captured well this subpopulation but failed to address all the RFs with low $n_x$ and $n_y$ values. Also, the binary model produced eight units with outlier $n_x$ and $n_y$ values — situated along the Cartesian axes — that correspond to atypical V1 RFs. With more time the fits would have been checked in-case they were pathological in some manner.

## 3.5 Relationship between spatial and temporal tuning properties of RFs

We also measured the correlation between the neuron's preferred spatial frequency and preferred temporal frequency. This was done by first converting the spatial dimensions of the 3D-spatio-temporal RF to a single dimension, which was done by summing the RF along the length of it's bars (see Singer et al. (2018) for methods). This provided a 2D-spatio-temporal RF to which was applied a 2D-Fourier transform. The peak in the Fourier power spectrum provided the preferred spatial frequency and temporal frequency of a unit. We found an inverse correlation between the preferred spatial frequency of units and their preferred temporal frequency, in both standard ($r^2$ = -0.45, $p < 10^{-6}$, n = 122) and binary ($r^2$ = -0.32, p = 0.0013, n = 100) models, consistent with what has been seen in the neurobiology (Deangelis et al., 1993).

## 3.6 A spiking temporal prediction model

Having observed that the temporal prediction model with binary outcome in the hidden layer — resembling spikes — generates a richer variety of RFs than the standard model, including non-canonical V1 RFs, we wondered whether a temporal prediction model with

integrate-and-fire units would be able to also reproduce these results. Here the spatial dimensions are encoded in the linear mapping from the input, whereas the temporal dimension relies purely on the recurrent integrative capacity of the units. We first built a model of Izhikevich's integrate-and-fire neurons in Tensorflow and proved that the units respond to the sequence of frames in the videos (Figure 9). Then we trained them by trying to predict the next frame given the previous one — and the history of the video encoded in the temporal dynamics of the units — by making use of a sigmoid surrogate gradient to overcome the binary nature of spikes.

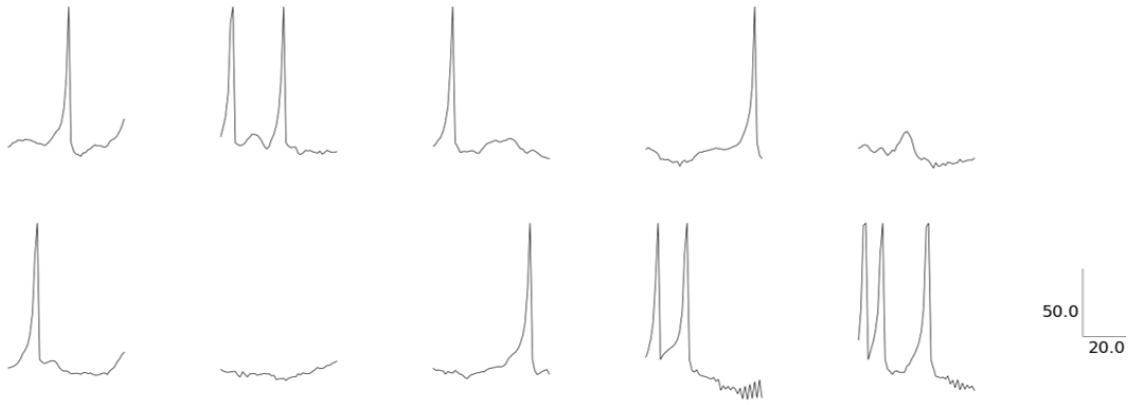


Figure 9: **Representative responses of 10 neurons to one of the videos after one training epoch.** Scale bar in mV/ms.

While the model did successfully produce spikes (Figure 9), the model did not respond as expected to the training process: Units started the training process firing several times along the simulation time and converged to a single spike after some epochs. From then on only the output weight and bias of the model evolved to sharpen the error in the cost function. Ideally other training options and model modifications would have been further explored, including changing Izhikevich's model for a simpler integrate-and-fire model or modifying the nature of the surrogate gradient. However, owing to the limited duration of this project this was not possible.

# 4 Discussion

This study has reported a feedforward convolutional network with binary units tuned to predict the close future of video frames by learning a sparse code. Here we have investigated some of the spatial properties of RFs emerging from binary units and compared them to the ones from the standard temporal prediction model. Convolutional temporal prediction models trained to predict on natural movies produce RFs that resemble in a number of ways RFs in V1. However we found strong differences between the standard and binary models. Whereas the standard model produces RFs with a greater number of on/off regions and more elongated bars, the binary model produces RFs which resemble better the ones seen in experimental data. Indeed, the binary model generates a greater diversity of RFs capturing atypical V1 RFs such as the centre-surround units observed in experimental data (Zylberberg et al., 2011; Cossell et al., 2015).

## 4.1 Strengths and limitations of the model

The clips used for training these temporal prediction models span over a time of 200 milliseconds. Here we have shown that predictions are drawn from information contained in the last 80 ms, which is consistent with conduction delays in cortical areas and rapid motor responses (Bixler et al., 1967). This way the visual system may overcome delays originated due to the time required to process sensory information of the present by making predictions about the near future based on sensory information from the last tens of milliseconds.

Due to time constraints the binary model could not be fully explored. Ideally, we would have chosen the regularisation strength by best prediction error on the validation set, however we observed that the prediction error on the validation set did not depend smoothly on the hyperparameters, rendering it difficult to judge where the true minimum was. This could be a consequence of the normalisation of the data used, that the validation

set was too small and maybe dominated by a subset of clips, or because the network did not train fully. Given more time, we would explore these considerations and choose the hyperparameters using a validation set. To enable the reader to see the influence of the hyperparameters on the resulting RFs we show in the appendix the set of RFs for a range of settings (Supplementary Figures S4 - S11). A more clear minimum could have been found if we had explored higher values of the scale of the surrogate gradient. Also, both standard and binary models could have been improved by exploring a wider range of regularisation strengths.

Although we ran the models for 200 epochs, it can be seen that they did not reach an absolute minimum (Figure S3). Therefore, running the binary model for longer could potentially improve the shapes of the generated RFs. Notably, the quantitative analysis was only made for the best hyperparameters, but this same pipeline could have been applied to sub-optimal values of the parameters in order to understand whether optimising for prediction actually yields the most realistic RFs. Finally, a further modification of the binary model would be a probabilistic binary network where binary units would have stochastic firing thresholds (Srinivasan and Roy, 2019).

## 4.2 The spiking non-linearity

Gradient descent through backpropagation is the algorithm of choice when training regular artificial neural networks. While methods to translate regular artificial neural networks into spike-based neural networks exist, neural networks based on spike timing are much more difficult to train. One of the main reasons for this is the non-differentiability of the spiking non-linearity. A spike function has a derivative which is zero everywhere except at the threshold value, where it is not defined. This property produces vanishing gradients during standard backpropagation expanding from the spiking units backwards to the input units. And because of the multiplicative properties of the chain rule, everything 'before' the spiking gets a zero derivative, therefore setting an end to the learning process.

The same problem happens with binary networks, in which units do not recurrently integrate activity across time, but they have a spiking non-linearity and a spiking threshold, and therefore also suffer from vanishing gradients. Interestingly, spiking neural networks can be formulated as recurrent neural networks, allowing training methods developed for the latter to be used within the former (Neftci et al., 2019). One common way to train recurrent neural networks is unrolling the network in time during backpropagation, a process sometimes referred to as backpropagation through time. This offers a manifold of methods to train both binary and spiking networks. Here we have trained a binary network by making use of a surrogate gradient derived from a scaled sigmoid (Zenke and Ganguli, 2018), which here we demonstrate shows promising results when applied to temporal prediction. Surrogate gradients are simple ways of transforming a discrete function into a continuous, differentiable function. Importantly, there are other functions which have been described to serve the purpose of surrogacy, including a quadratic truncated function (Bohte, 2011) — resulting in a derivative of the type of rectifying linear units — and exponential functions (Shrestha and Orchard, 2018). However, this is just one of the possible alternatives to conventional training, and the number of methods available to train spiking networks is rapidly increasing. Although here we failed to develop a spiking version of the temporal prediction model with a surrogate sigmoid function, here we want to state that this model is, indeed, still at early stages, yet to be fully explored and developed. We actually managed to train the model in the sense that we overcome the spiking non-linearity and the units evolved along epochs, but they converged to a solution that was far from what we expected. Further research should yield improvements of the model by simplifying the quadratic integrate-and-fire model to a standard, simpler, conductance-based integrate-and-fire model, or by designing a slightly different architecture of the network (e.g. varying how the input is feed to the spiking units), or by using other kinds of surrogate functions.

One big concern about propagating the error in the output with gradient descent through backpropagation is that this algorithm does not seem to be plausible in biological systems

(Stork, 1989). In contrast, local learning rules in which errors are only propagated from the pre- and post- synaptic cells, but not from a far output, seem to be a more appropriate. Recent research has uncovered new methods to efficiently train deep neural networks with local rules (Mostafa et al., 2018). Therefore, future research should not only aim at exploring temporal prediction with spiking units, but also at training these networks with local plasticity rules.

## Conclusion

Temporal prediction has been suggested as a very general principle behind sensory processing in the brain. Indeed optimising networks for temporal prediction has been shown to reproduce many aspects of neural tuning in retina (Singer et al., 2019), primary visual cortex including simple (Singer et al., 2018, 2019; Chalk et al., 2017) and complex cells (Singer et al., 2019), primary auditory cortex (Singer et al., 2018), and higher visual areas (Singer et al., 2019). However, a limitation of all these models is that they do not use binary units — reminiscent of neural spikes — but instead real valued units, more reflective of firing rates. Here we demonstrate that this need not be a problem for this hypothesis, that indeed binary temporal prediction models can reproduce simple cell properties when trained on natural movies just as can standard temporal prediction models. Furthermore, certain aspects not captured by the standard temporal prediction model such as centre-surround receptive fields are captured by the binary model. Development of hierarchical and recurrent versions of the binary temporal prediction model may hold many more interesting features to compare to the brain.

# 5 References

Abadi, M., Barham, P., Chen, J., Chen, Z., Davis, A., Dean, J., Devin, M., Ghemawat, S., Irving, G., Isard, M., Kudlur, M., Levenberg, J., Monga, R., Moore, S., Murray, D. G., Steiner, B., Tucker, P., Vasudevan, V., Warden, P., Wicke, M., Yu, Y., and Zheng, X. (2016). Tensorflow: A system for large-scale machine learning. In *12th USENIX Symposium on Operating Systems Design and Implementation (OSDI 16)*, pages 265–283.

Adelson, E. H. and Bergen, J. R. (1985). Spatiotemporal energy models for the perception of motion. *J. Opt. Soc. Am. A*, 2(2):284–299.

Barlow, H. (2001). Redundancy reduction revisited. *Network: Computation in Neural Systems*, 12(3):241–253.

Berkes, P., Turner, R. E., and Sahani, M. (2009). A structured model of video reproduces primary visual cortical organisation. *PLOS Computational Biology*, 5(9):1–16.

Berkes, P. and Wiskott, L. (2005). Slow feature analysis yields a rich repertoire of complex cell properties. *Journal of Vision*, 5(6):9–9.

Bialek, W., Nemenman, I., and Tishby, N. (2001). Predictability, complexity, and learning. *Neural Computation*, 13(11):2409–2463.

Bohte, S. M. (2011). Error-backpropagation in networks of fractionally predictive spiking neurons. In Honkela, T., Duch, W., Girolami, M., and Kaski, S., editors, *Artificial Neural Networks and Machine Learning – ICANN 2011*, pages 60–68, Berlin, Heidelberg. Springer Berlin Heidelberg.

Carlson, N. L., Ming, V. L., and DeWeese, M. R. (2012). Sparse codes for speech predict spectrotemporal receptive fields in the inferior colliculus. *PLOS Computational Biology*, 8(7):1–15.

Chalk, M., Marre, O., and Tkačik, G. (2017). Toward a unified theory of efficient,

predictive, and sparse coding. *Proceedings of the National Academy of Sciences*, 115(1):186–191.

Cossell, L., Iacaruso, M. F., Muir, D. R., Houlton, R., Sader, E. N., Ko, H., Hofer, S. B., and Mrsic-Flogel, T. D. (2015). Functional organization of excitatory synaptic strength in primary visual cortex. *Nature*, 518(7539):399–403.

Deangelis, G. C., Ohzawa, I., and Freeman, R. D. (1993). Spatiotemporal organization of simple-cell receptive fields in the cats striate cortex. i. general characteristics and postnatal development. *Journal of Neurophysiology*, 69(4):1091–1117.

Hateren, J. H. V. and Schaaf, A. V. D. (1998). Independent component filters of natural images compared with simple cells in primary visual cortex. *Proceedings of the Royal Society of London. Series B: Biological Sciences*, 265(1394):359–366.

Heeger, D. J. (2017). Theory of cortical function. *Proceedings of the National Academy of Sciences*, 114(8):1773–1782.

Helmholtz, H. v. (1924). Concerning the perceptions in general. In *Treatise on physiological optics*, volume 3. Dover, New York.

Hubel, D. H. and Wiesel, T. N. (1959). Receptive fields of single neurones in the cat's striate cortex. *The Journal of Physiology*, 148(3):574–591.

Izhikevich, E. (2003). Simple model of spiking neurons. *IEEE Transactions on Neural Networks*, 14(6):1569–1572.

Jones, J. P. and Palmer, L. A. (1987a). An evaluation of the two-dimensional gabor filter model of simple receptive fields in cat striate cortex. *Journal of Neurophysiology*, 58(6):1233–1258.

Jones, J. P. and Palmer, L. A. (1987b). The two-dimensional spatial structure of simple receptive fields in cat striate cortex. *Journal of Neurophysiology*, 58(6):1187–1211. PMID: 3437330.

Kingma, D. P. and Ba, J. (2014). Adam: A Method for Stochastic Optimization. *arXiv e-prints*, page arXiv:1412.6980.

Kreile, A. K., Bonhoeffer, T., and Hübener, M. (2011). Altered visual experience induces instructive changes of orientation preference in mouse visual cortex. *Journal of Neuroscience*, 31(39):13911–13920.

Mostafa, H., Ramesh, V., and Cauwenberghs, G. (2018). Deep supervised learning using local errors. *Frontiers in Neuroscience*, 12:608.

Neftci, E. O., Mostafa, H., and Zenke, F. (2019). Surrogate Gradient Learning in Spiking Neural Networks. *arXiv e-prints*, page arXiv:1901.09948.

Niell, C. M. and Stryker, M. P. (2008). Highly selective receptive fields in mouse visual cortex. *Journal of Neuroscience*, 28(30):7520–7536.

Ohzawa, I., Deangelis, G. C., and Freeman, R. D. (1996). Encoding of binocular disparity by simple cells in the cats visual cortex. *Journal of Neurophysiology*, 75(5):1779–1805.

Olshausen, B. A. and Field, D. J. (1996). Emergence of simple-cell receptive field properties by learning a sparse code for natural images. *Nature*, 381(6583):607–609.

Olshausen, B. A. and Field, D. J. (1997). Sparse coding with an overcomplete basis set: A strategy employed by v1? *Vision Research*, 37(23):3311–3325.

Palmer, S. E., Marre, O., Berry, M. J., and Bialek, W. (2015). Predictive information in a sensory population. *Proceedings of the National Academy of Sciences*, 112(22):6908–6913.

Reid, R. C., Soodak, R. E., and Shapley, R. M. (1987). Linear mechanisms of directional selectivity in simple cells of cat striate cortex. *Proceedings of the National Academy of Sciences*, 84(23):8740–8744.

Ringach, D. L. (2002). Spatial structure and symmetry of simple-cell receptive fields in macaque primary visual cortex. *Journal of Neurophysiology*, 88(1):455–463.

Ringach, D. L. (2004). Mapping receptive fields in primary visual cortex. *The Journal of Physiology*, 558(3):717–728.

Salisbury, J. M. and Palmer, S. E. (2016). Optimal prediction in the retina and natural motion statistics. *Journal of Statistical Physics*, 162(5):1309–1323.

Shannon, C. E. (1948). A mathematical theory of communication. *Bell System Technical Journal*, 27(3):379–423.

Shrestha, S. B. and Orchard, G. (2018). Slayer: Spike layer error reassignment in time. In Bengio, S., Wallach, H., Larochelle, H., Grauman, K., Cesa-Bianchi, N., and Garnett, R., editors, *Advances in Neural Information Processing Systems 31*, pages 1412–1421. Curran Associates, Inc.

Singer, Y., Teramoto, Y., Willmore, B. D., Schnupp, J. W., King, A. J., and Harper, N. S. (2018). Sensory cortex is optimized for prediction of future input. *eLife*, 7.

Singer, Y., Willmore, B. D. B., King, A. J., and Harper, N. S. (2019). Hierarchical temporal prediction captures motion processing from retina to higher visual cortex. *bioRxiv*.

Srinivasan, G. and Roy, K. (2019). Restocnet: Residual stochastic binary convolutional spiking neural network for memory-efficient neuromorphic computing. *Frontiers in Neuroscience*, 13:189.

Stork, D. G. (1989). Is backpropagation biologically plausible? *International Joint Conference on Neural Networks*.

Sutton, R. S. and Barto, A. G. (1981). An adaptive network that constructs and uses an internal model of its world. *Cognition and Brain Theory*, 4:217–246.

van Hateren, J. H. and Ruderman, D. L. (1998). Independent component analysis of natural image sequences yields spatio-temporal filters similar to simple cells in primary visual cortex. *Proceedings of the Royal Society of London. Series B: Biological Sciences*, 265(1412):2315–2320.

Zenke, F. and Ganguli, S. (2018). Superspike: Supervised learning in multilayer spiking neural networks. *Neural Computation*, 30(6):1514–1541.

Zylberberg, J., Murphy, J. T., and Deweese, M. R. (2011). A sparse coding model with synaptically local plasticity and spiking neurons can account for the diverse shapes of v1 simple cell receptive fields. *PLoS Computational Biology*, 7(10).

# 6 Supplementary figures

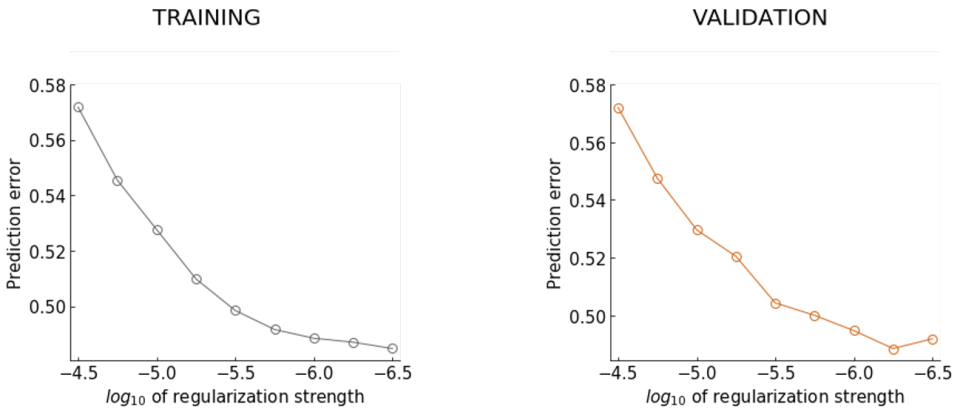


Figure S1: **Tuning regularisation strength of the standard model.** Prediction error in the output measured as defined in equation (3) (left: training set; right: validation set) after 50 training epochs as a function of the regularisation strength (logarithmic units).

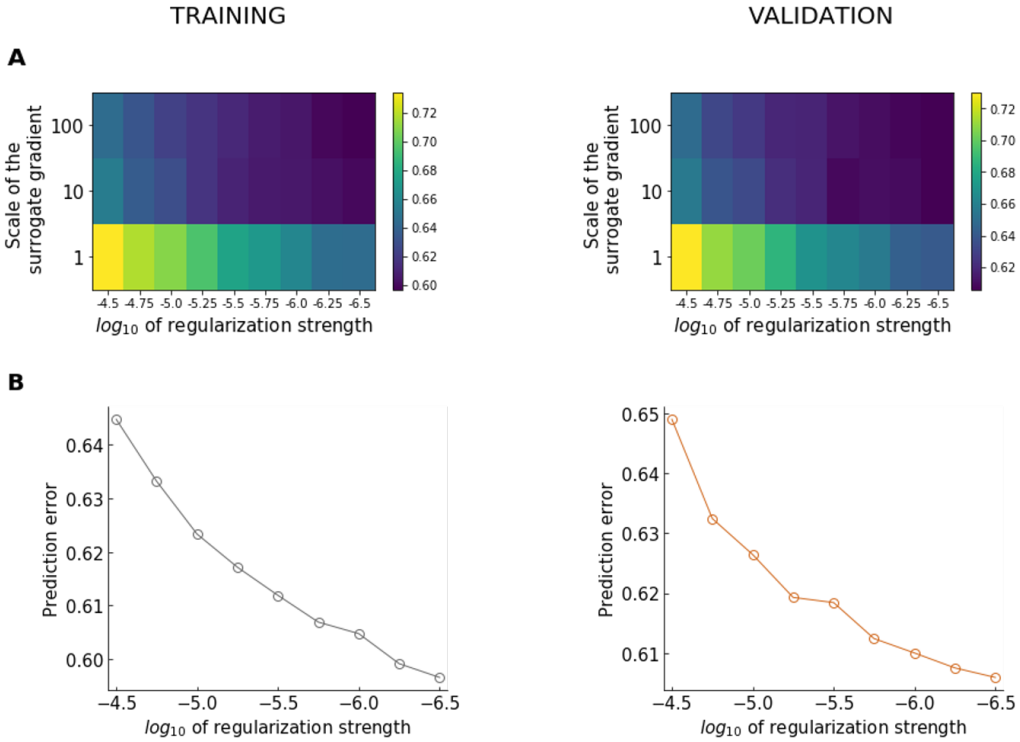


Figure S2: **Hyperparameter search for the binary model.** (A) Heatmat representing prediction error in the output measured as defined in equation (3) (left: training set; right: validation set) after 50 training epochs as a function of the regularisation strength (logarithmic units) and the scale of the surrogate gradient. Colour represents prediction error, with scale bar on the right. (B) Prediction error for a fixed scale of the surrogate gradient of 100 (best prediction).

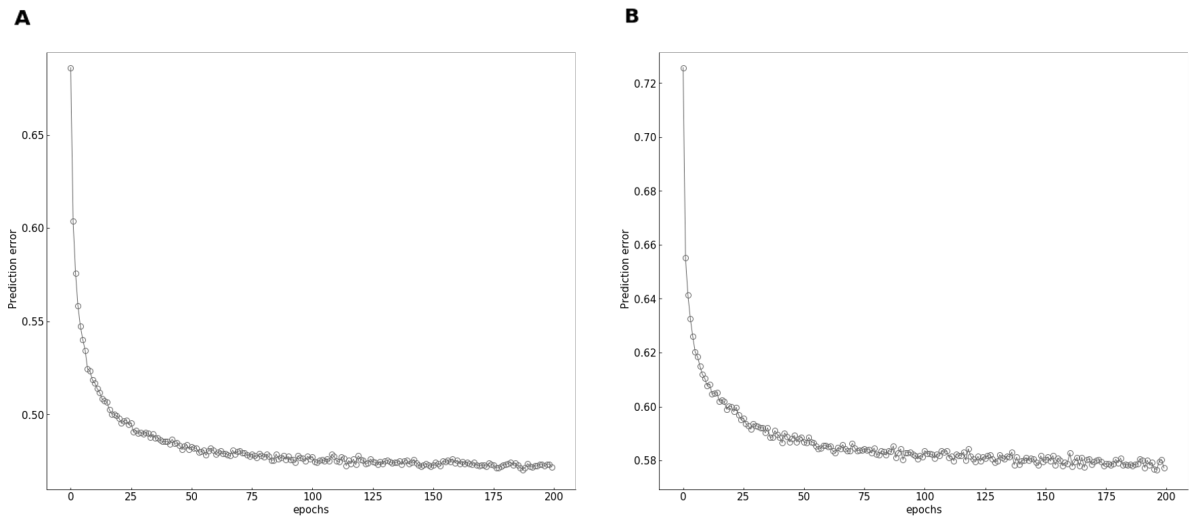


Figure S3: **Learning curve for standard (A) and binary (B) models when tuned to their best hyperparameters**. Prediction error measured as the mean squared error of the training dataset.

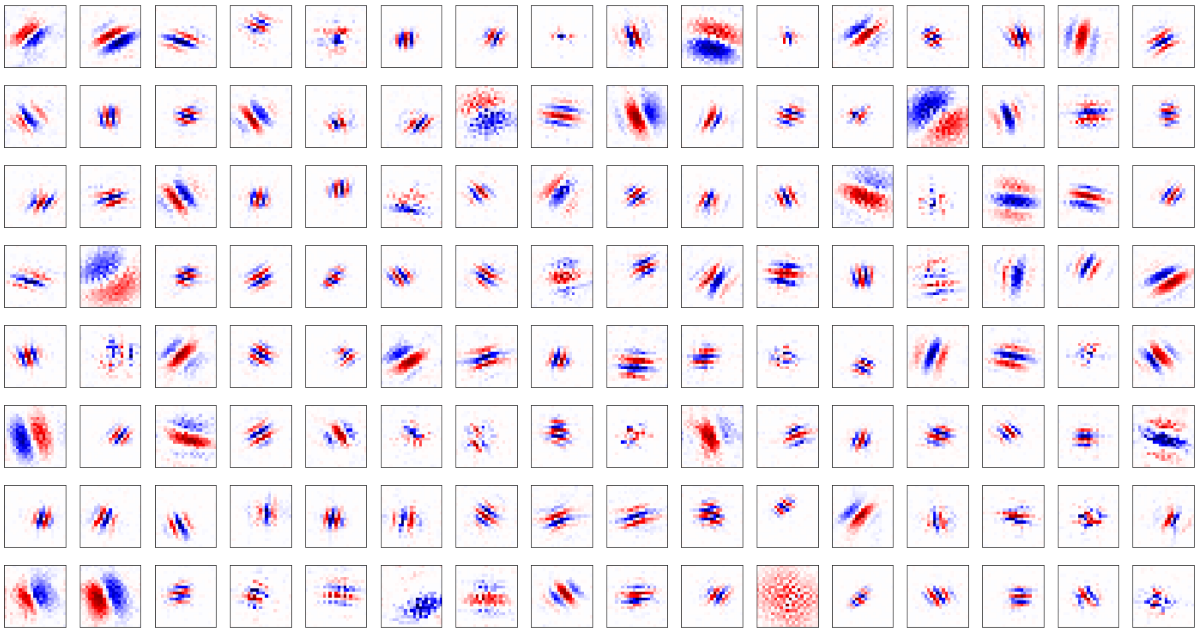

Figure S4: **Complete set of receptive fields generated by the standard model at their highest power for $\lambda = 10^{-4.5}$.** Colour represents 'on' and 'off' regions.

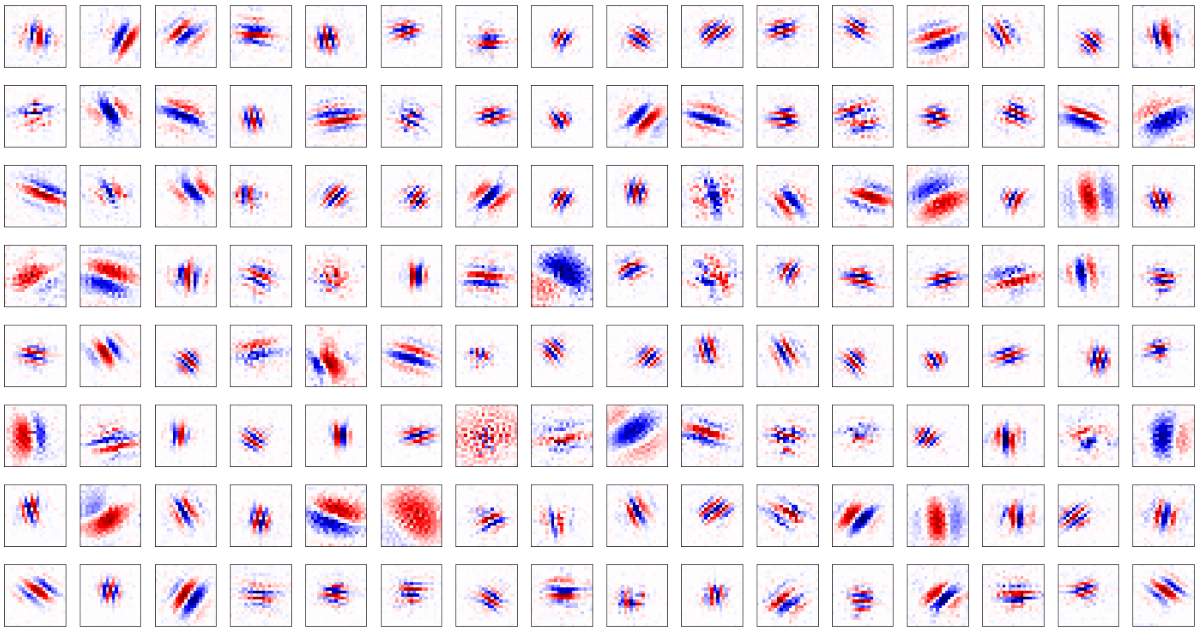

Figure S5: **Complete set of receptive fields generated by the standard model at their highest power for $\lambda = 10^{-4.75}$.** Colour represents 'on' and 'off' regions.

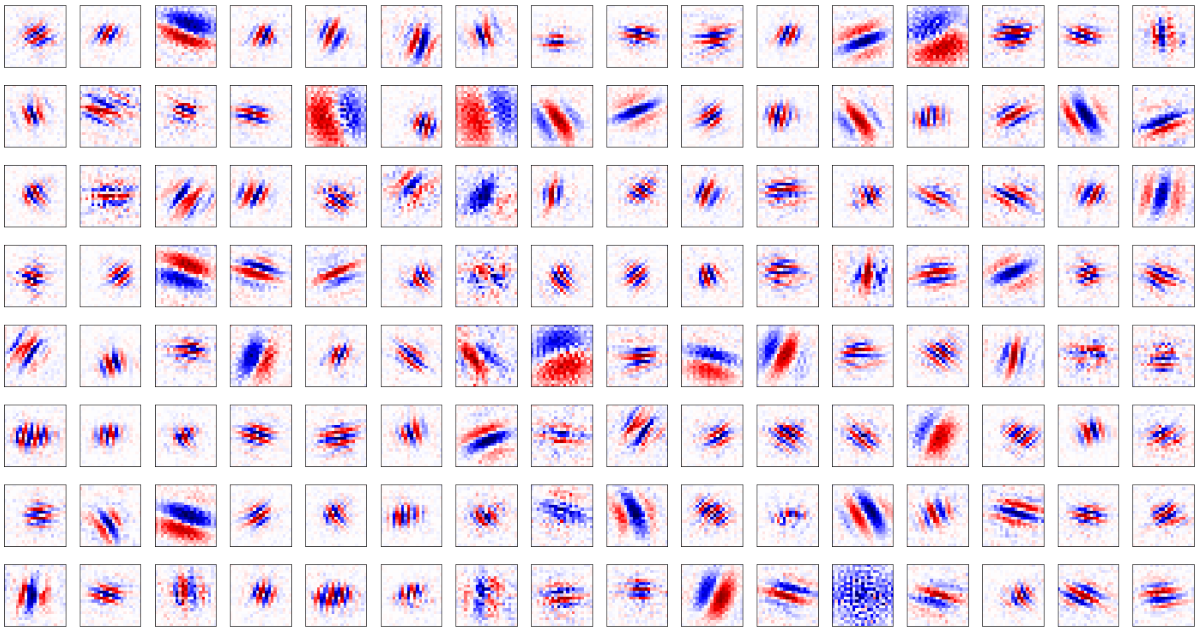

Figure S6: **Complete set of receptive fields generated by the standard model at their highest power for $\lambda = 10^{-5.25}$.** Colour represents 'on' and 'off' regions.

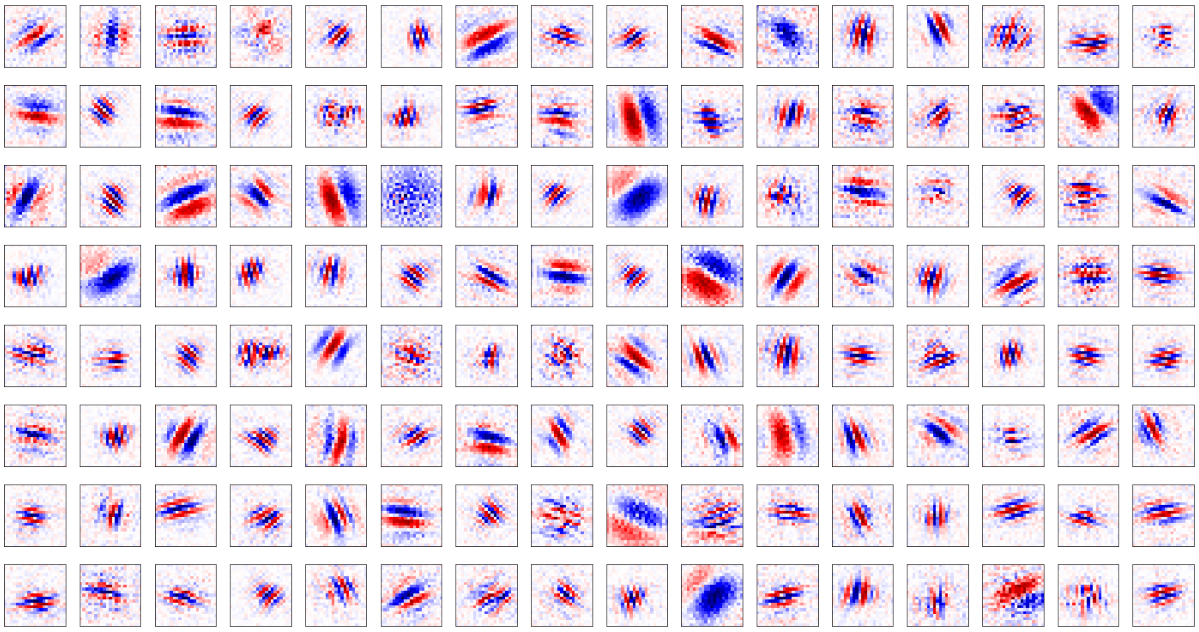

Figure S7: **Complete set of receptive fields generated by the standard model at their highest power for $\lambda = 10^{-5.5}$.** Colour represents 'on' and 'off' regions.

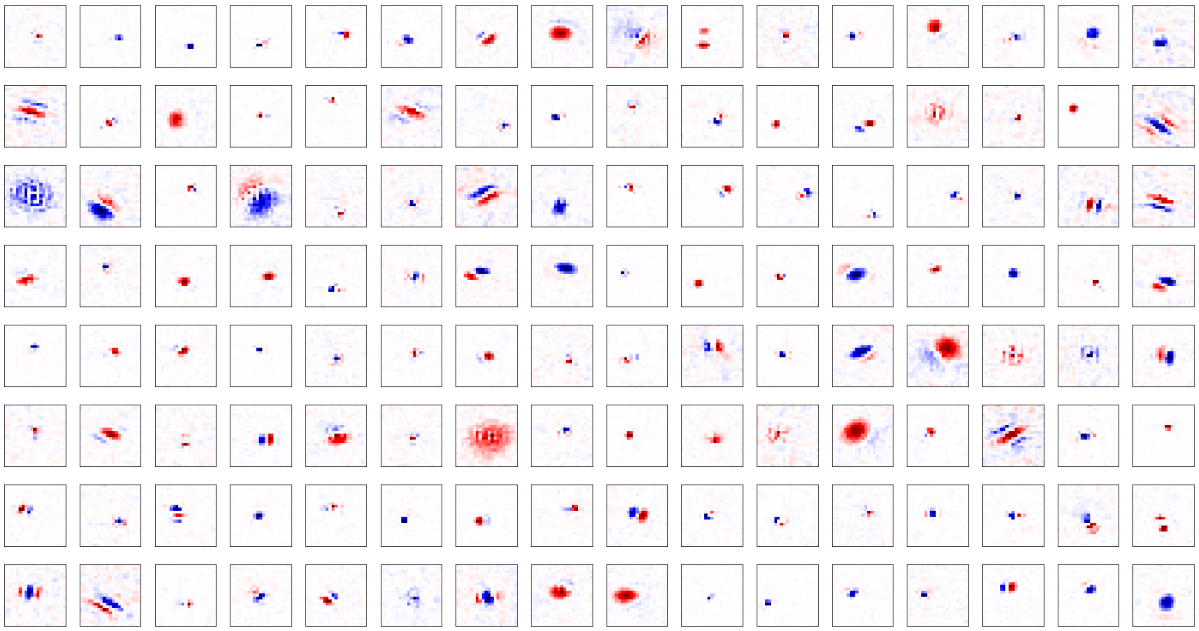

Figure S8: **Complete set of receptive fields generated by the binary model at their highest power for $\lambda = 10^{-5.25}$.** Colour represents 'on' and 'off' regions.

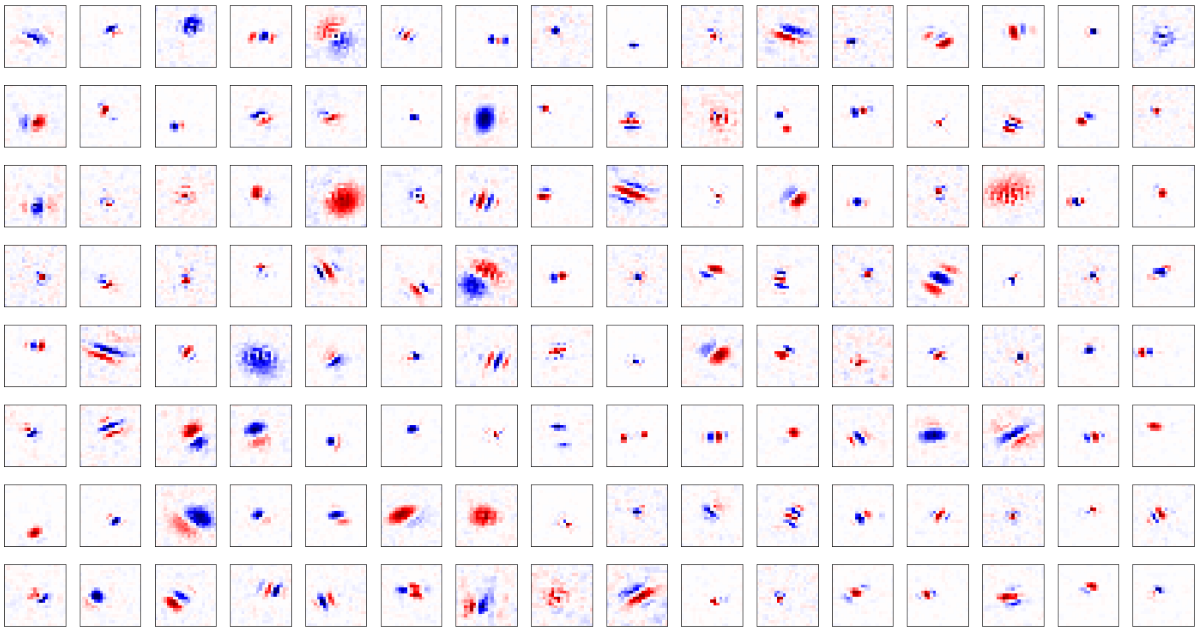

Figure S9: **Complete set of receptive fields generated by the binary model at their highest power for $\lambda = 10^{-5.5}$.** Colour represents 'on' and 'off' regions.

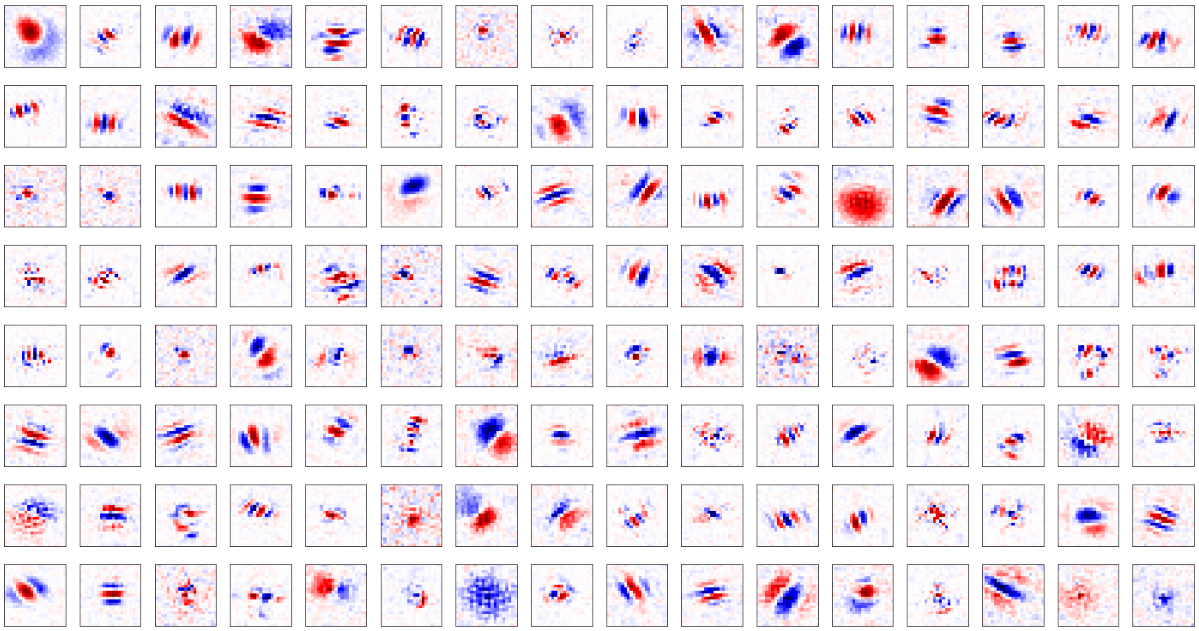

Figure S10: **Complete set of receptive fields generated by the binary model at their highest power for $\lambda = 10^{-6}$.** Colour represents 'on' and 'off' regions.

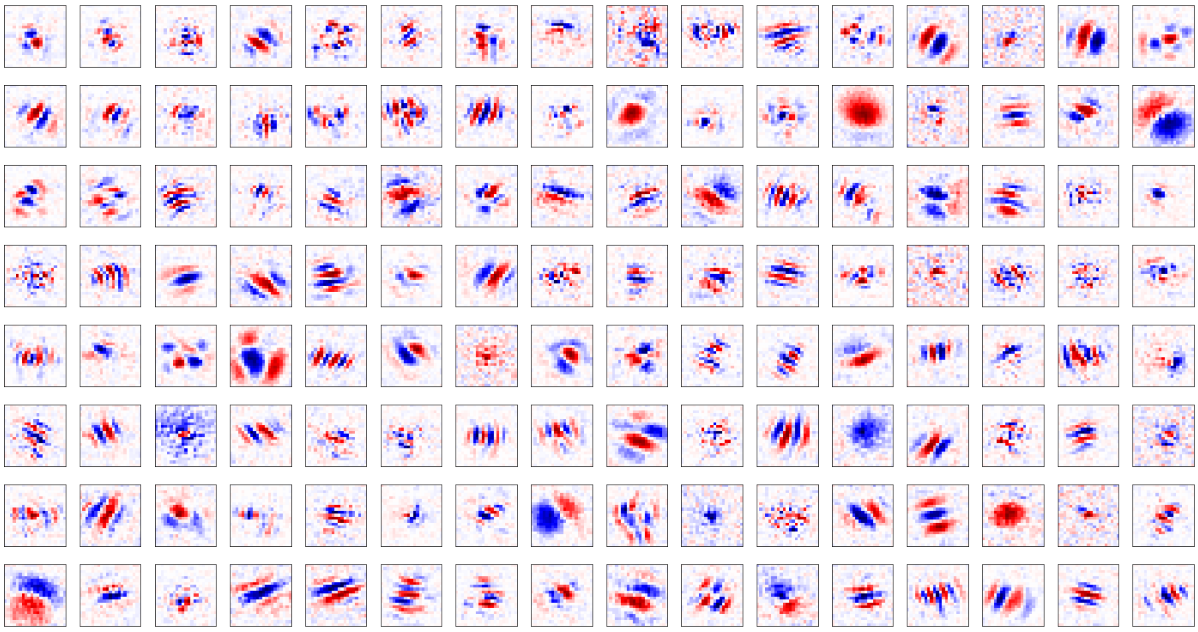

Figure S11: **Complete set of receptive fields generated by the binary model at their highest power for $\lambda = 10^{-6.25}$.** Colour represents 'on' and 'off' regions.