A model-free approach to link neural activity to behavioral tasks

Jamie D Costabile[1,2,5], Kaarthik A Balakrishnan[1,3,5], Sina Schwinn[1,4] and Martin Haesemeyer*[1]

[1]Department of Neuroscience, The Ohio State University College of Medicine, Columbus, OH, USA
[2]Present address: Hitachi Solutions America, Ltd. Irvinve, CA, USA
[3]Interdisciplinary Biophysics Graduate Program, Columbus, OH, USA
[4]Present address: Mass Eye and Ear, Massachusetts General Brigham, Boston, MA, USA
[5]*These authors contributed equally*

Summary

Brains are not engineered solutions to a well-defined problem but arose through selective pressure acting on random variation. It is therefore unclear how well a model chosen by an experimenter can relate neural activity to experimental conditions. Here we developed "Model-free identification of neural encoding (MINE)" using convolutional neural networks (CNN) to relate aspects of tasks to neural activity. Although flexible, CNNs are difficult to interpret. We use Taylor decomposition approaches and explainable machine learning techniques to understand the mapping from task features to activity. We apply MINE to a published cortical dataset as well as experiments designed to probe thermoregulatory circuits in zebrafish. MINE allowed us to characterize neurons according to their receptive field and computational complexity, features which anatomically segregated in the brain. We also identified a new class of neurons that integrate thermosensory and behavioral information which eluded us previously when using traditional clustering and regression-based approaches.

Introduction

Contemporary neuroscience generates large datasets of neural activity in behaving animals[1–3]. To gain insight from these large-scale recordings, it is often desired to identify neurons with activity related to the behavioral task at hand. A common approach to this problem is to intuit a functional form ("a model") by which predictors such as sensory stimuli, motor actions and internal states might be related to neural activity. Neurons can subsequently be classified into those with activity explained by features of the task and the chosen model and those with activity likely to be unrelated or background noise. A simple yet powerful approach is to use linear regression to explain neural activity as the weighted sum of sensory and motor features recorded during the experiment[4–7]. Since linear regression can accommodate nonlinear transformations of input variables, this technique can encompass diverse relationships between predictors and neural activity. Similarly, more flexible solutions using basis functions[8,9] can be used to identify task related neurons. However, brains are not engineered solutions to a well-defined problem but arose through selective pressure acting on random variation[10–12]. It is therefore unclear how well neural activity can be captured by a well-defined function chosen by the experimenter as the test model. In theory these issues could be alleviated by increasing the flexibility of the function using a large set of parameters. However, this approach is fraught with hazard as model fits become less and less well constrained, leading to an explosion of variance and over-fitting to the training data[9,13].

Artificial neural networks (ANN) can in principle accommodate any mapping of predictors to neural activity[14,15]. At the same time, they can be designed to generalize well to untrained inputs[16,17]. Due to this flexibility, insights into nonlinear receptive fields of visual

*Correspondence: haesemeyer.1@osu.edu

and auditory neurons[18–22] and into the encoding of limb motion in somatosensory cortex have been gained using ANNs[23]. However, an obvious drawback of ANNs is that they are much harder to interpret than models based on intuition and data exploration.

Here we introduce "Model-free identification of neural encoding", (MINE), an approach that uses simple convolutional neural networks (CNN) to learn mappings from predictors (stimuli, behavioral actions, internal states) to neural activity. By employing Taylor expansion approaches and techniques used in interpretable machine learning such as the nonlinearity coefficient, we gain insight into how the CNN uses information about stimuli, states and behaviors to predict neural activity. This allows us to make statements about a) how strongly individual predictors contribute to the activity of individual neurons, b) whether the mapping is likely linear or non-linear and c) the structure of the neuron's receptive field. Aside from network architecture and initialization conditions MINE is essentially model-free. Furthermore, by incorporating a convolutional layer, temporal or spatial transformations of inputs introduced by the technique (such as calcium indicator effects) or by the brain (such as differentiation of a signal, edge detection) will be learned seamlessly by the model and do not have to be captured through a priori transformations of the task variables.

Here we demonstrate the utility of MINE using a ground-truth dataset and a cortical mouse widefield imaging dataset. We then designed a set of experiments to exhaustively probe thermoregulatory circuits in larval zebrafish. Specifically, we exploit the flexibility of MINE to provide randomly varied temperature stimuli across zebrafish and imaging planes while maintaining the ability to identify functional groups of neurons based on features of the trained CNNs. Using MINE, we discover a new functional class of neurons integrating thermosensory with behavioral information. Combining MINE with anatomical mapping, we also identify brain regions containing neuronal responses of increased computational complexity.

Results

A model-free approach to identify task-relevant neurons

MINE (for Model-free Identification of Neural Encoding) centers around convolutional neural networks (CNN) to overcome challenges with predefined models while maintaining interpretability. Feed-forward artificial neural networks are capable to approximate any function[14,15] and therefore afford great flexibility in capturing the relationship between "predictors" (sensory stimuli, behavioral actions, internal states, etc.) and the activity of individual neurons. We designed a simple three-layered network architecture (Figure 1A). It consists of a linear convolutional layer (to accommodate temporal transformations such as calcium indicator effects) and two dense layers that capture nonlinear transformations. The architecture's simplicity speeds up training and eases interpretation while capturing transformations across time (i.e. the convolutional layers) and nonlinear effects (i.e. dense layers with nonlinear activations functions). We chose a continuously differentiable activation function for our dense network layers. Unlike using the popular ReLu nonlinearity, this allows us to calculate higher-order derivatives which capture interaction effects and quantify the nonlinearity of transformations. Most network hyperparameters including the specific activation function ("swish"[24]) were determined by minimizing test error on a small dataset (see Methods). While this parameter can be easily adjusted, we chose a fixed length of 10 seconds for our convolutional filters to capture temporal effects in all subsequent experiments.

To test MINE's ability to capture neural transformations and encoding of predictors we generated a ground-truth dataset. This dataset consists of four randomly generated predictors (Figure 1B). Two of these, S1 and S2, vary continuously in time mimicking sensory stimuli. The other two, M1 and M2, are discrete like motor or decision variables. From these predictors we generated "neural responses" that depend on single predictors

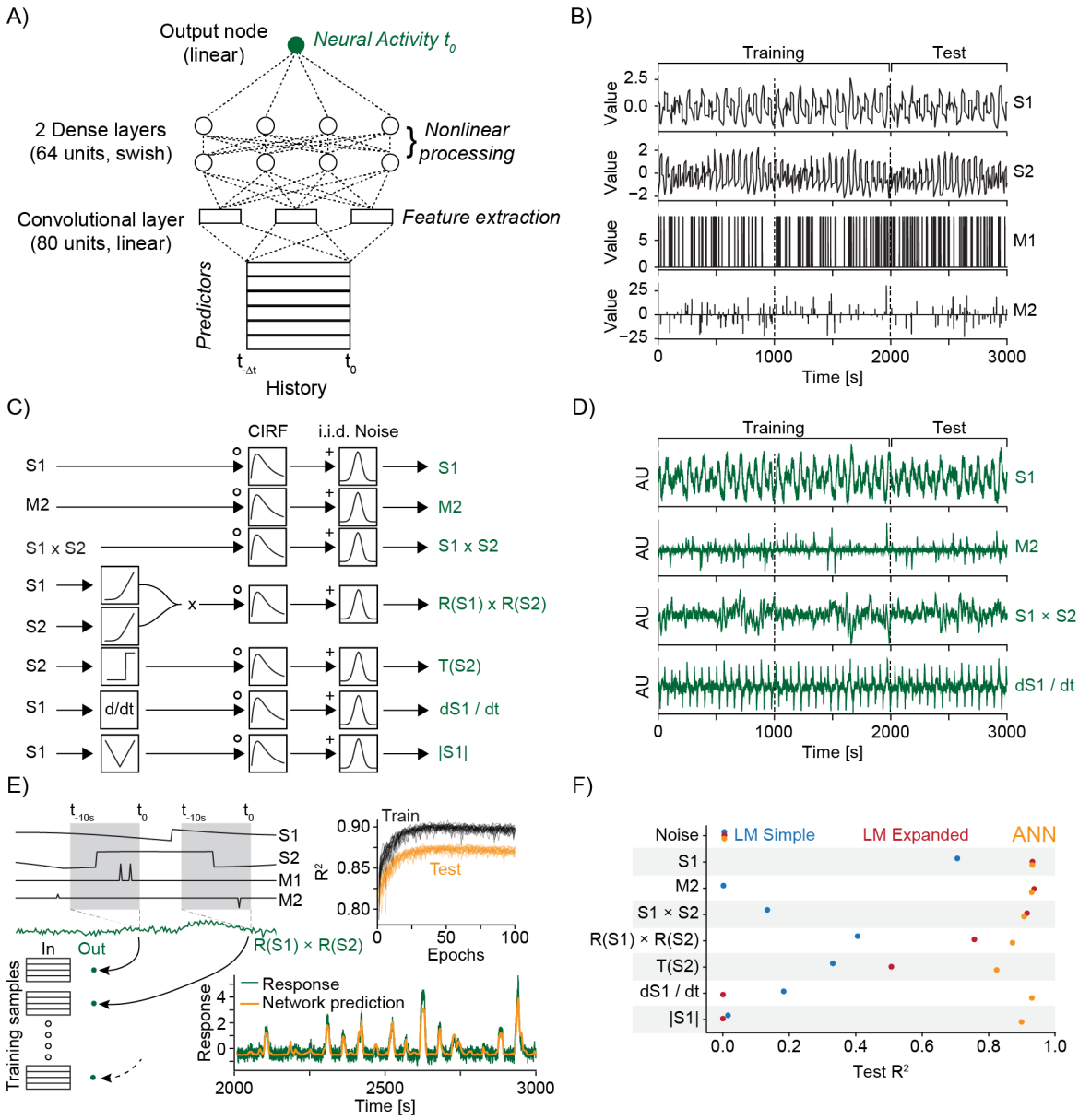

Figure 1: MINE accomodates a large set of predictor-activity relationships

A) Schematic of the convolutional neural network used.

B) The predictors that make up the ground-truth dataset. S1 and S2 are continuously variable predictors akin to sensory variables while M1 and M2 are discrete in time, akin to motor or decision variables. Dashed lines indicate a third each of the data with the first two thirds used for training of models and the last third used for testing.

C) Schematic representation of ground-truth response generation.

D) Example responses in the ground-truth dataset. Labels on the right refer to the response types shown in C).

E) The model predicts activity at time t using predictors across time $t - \Delta t$ to t as inputs. The schematic shows how this is related to the generation of training and test data. Top inset shows development of training and test error across training epochs for 20 models trained on the R(S1)xR(S2) response type. Bottom inset shows example prediction (orange) overlaid on response (dark green).

F) Fraction of explained variance on test data for a simple linear regression model (blue), a linear regression model including first-order interaction terms and calcium kernel convolution (red) as well as the CNN model (orange). Each dot represents the average across 20 models. While the standard deviation is represented by a dash, it is smaller than the dot size in all cases and therefore not visible in the graph.

or interactions with and without intervening nonlinear transformations (Figure 1C and S1A). We added gaussian noise to all neural responses after convolution with a calcium kernel to approach conditions that might be ob-

served in functional calcium imaging experiments (Figure 1D). For each neuron in our ground-truth dataset we subsequently trained a CNN to predict the activity from the predictors (Figure 1E). To assess generalization, we split our dataset into a training set (two thirds of the data) and a test set (last third) (Figure 1B and D). Training for 100 epochs led to excellent predictions of "neural activity" while maintaining generalizability as assessed by the fraction of explained variance (R^2 value) of the test data (Figure 1E-F).

We sought to compare our ANN based approach to another widespread approach to model single-neuron activity, linear regression. Using the four predictors (Figure 1B) as inputs a simple linear regression model fails to explain activity in most cases (Figure 1F and S1C). This is expected, since the linear regression model cannot learn the dynamics of the calcium indicator unlike the CNN. We therefore constructed an alternative model in which we convolved the predictors with the known "calcium kernel" (Figure S1B). In this expanded linear model we also included all first-order interaction terms by including pairwise products between predictors (Figure S1B). As expected, this model matches the performance of the CNN in more response categories including non-linear interactions (Figure 1F). However, this is because the function it represents was designed using *a posteriori* knowledge about the responses. Nonetheless, the expanded linear model is poor in capturing nonlinear transformations of predictors and fails to capture responses that relate to the time-derivative of an input, e.g., as expected in adapting neurons (Figure 1F). This further illustrates the point that a model-based approach is limited to the constraints of the chosen model.

As shown, MINE can identify responses that depend on predictors independent of the linearity of these relationships. The underlying CNN is able to learn temporal transformations of inputs such as shifts in time or convolutions which otherwise have to be explicitly provided to a regression model[4,7]. For example, this removes the requirement of estimating calcium response kernels which might differ across different neurons. MINE can also identify responses that depend on derivatives of the input meaning that one predictor such as position can be used to identify neurons that depend on the velocity or acceleration of a stimulus.

Taylor expansion identifies predictors driving the response

Compared to regression models, MINE seemingly has a drawback: a lack of interpretability. Statistical methods can identify factors that significantly contribute to a regression model's output. Similarly, the magnitude of individual weights in models fit to data can give an idea of the importance of specific predictors. Corresponding overt parameters do not exist in the CNN model. Theoretically, it would be possible to re-fit successful CNN models in a stepwise manner, leaving out or adding in specific predictors[25]. However, since we are interested in uncovering relationships between predictors this could only succeed if all individual combinations between inputs are tested. This would be prohibitive for large sets of sensory and behavioral predictors as it would require repeatedly re-training all networks.

We therefore perform local Taylor expansions of the network at different experimental timepoints. In other words, we differentiate the network's learned transfer function that transforms predictors into neural activity. This allows us to transform the complex nonlinear CNN with interdependent parameters into a series of simple polynomial equations much like the expanded linear regression model. Since we chose a continuously differentiable activation function for our network we can compute the function's derivatives (first, second, third, etc.) of the output with respect to any predictor and timepoint in the input. We specifically focus on the first-order derivatives (the *Jacobian* J, i.e. single-predictor influences) and the matrix of second-order derivatives (the *Hessian* H, i.e. pairwise interactions between predictors) . Since the decomposition is a sum of terms that depend on individual inputs or their combinations, we can use it to determine how important individual predictors, such as sensory stimuli, are in shaping the neu-

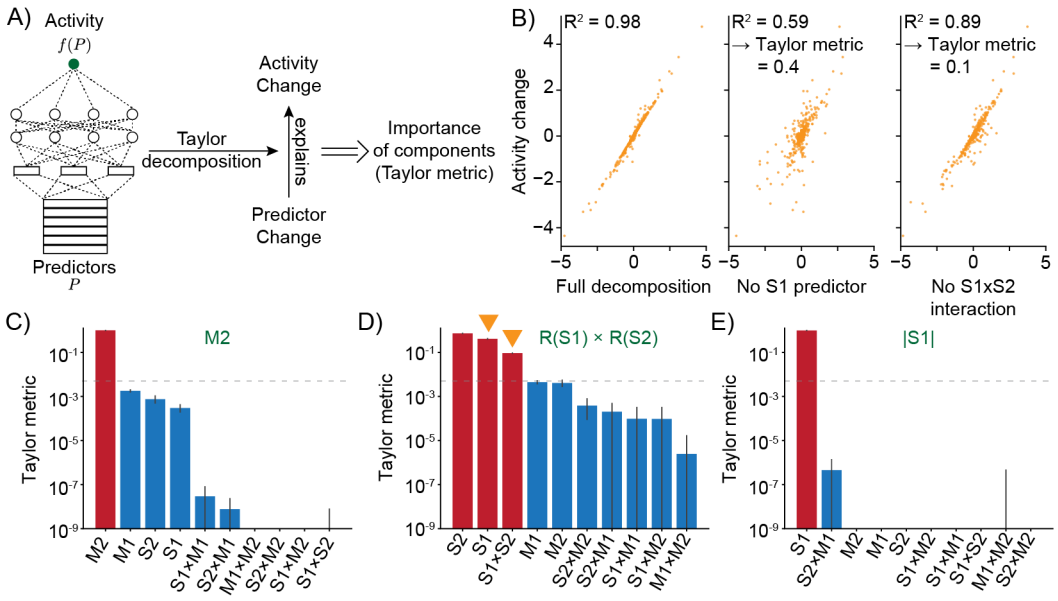

Figure 2: Taylor decomposition reveals contributing single factors and interactions

A) The neural model translates predictors into neural activity. By computing the vector of first order derivatives (Jacobian J) and matrix of second-order derivatives (Hessian H) the function implemented by the CNN can be linearized locally. Relating changes in predictors to changes in activity for full and partial linearizations reveals those predictors and interactions that contribute to neural activity.

B) Example of Taylor metric computation. Left relationship between the CNN output and the full taylor approximation. Middle, after removal of the term that contains the S1 predictor. Right, after removal of the term that describes the interaction between S1 and S2.

C-E) three examples responses and associated Taylor metrics. Red bars indicate predictors that are expected to contribute, blue bars those that should not contribute. Dashed line indicates a threshold at $5x10^{-3}$ which identifies true contributing predictors in all cases. Error bars are 95 % bootstrap confidence intervals.

C) M2 response type.

D) R(S1) x R(S2) response type. Arrowheads indicate the metrics which are shown in the example (right and middle) of B).

E) Response type encoding the absolute value of S1.

ral response (Figure 2A and Figure S2A). Across our ground-truth dataset we find a high correlation between the change in output of the Taylor expansion and the true change in network output (Figure 2B, left panel) indicating that the local expansions using first and second order derivatives sufficiently capture the relationship between predictors and neural activity. Importantly, decomposition is a way to efficiently test the importance of different predictors in contributing to network output and hence neural activity. We calculate a "Taylor metric" score which measures the fraction of explanatory power lost when terms are removed from the full Taylor expansion (Figure 2B).

On our ground-truth dataset the Taylor metric correctly identifies the contributing predictors and their interactions. Sorting individual terms by this metric consistently ranks those that we expect to contribute (Figure 2C-E, red bars) higher than those which should not (Figure 2C-E, blue bars). This is true both for individual predictors as well as interaction terms in the case where the response depends on the product of inputs. Moreover, we can empirically set a threshold of the Taylor metric ($5x10^{-3}$) which separates the expected contributing terms from the others across our ground-truth dataset in all but one case (Figure 2C-E; Figure S2B-H).

In summary, MINE was able to correctly identify contributions of predictors, such as sensory stimuli or behavioral actions to neural responses by local expansions of the trained CNNs. MINE also correctly identifies nonlinear interactions in generating the neural responses on our ground-truth dataset. This indicates that we can reduce the lack of interpretability of the CNN models

and approach the expressivity of linear regression models while maintaining the ability to model nonlinear transformations and interactions of task variables.

Disambiguating linear and non-linear processing

Linear computations are limited in their expressivity and it is generally believed that the computational power of the nervous system depends on nonlinear computation[26–28]. In spite of this, the responses of many neurons tend to depend almost linearly on sensory or behavioral features[4,7,29,30]. The latter idea aligns with the hypothesis that information important to the animal should be linearly decodable by neural circuits[10,31,32]. Disambiguating linear from nonlinear processing therefore provides important insight into circuit structure and function. Once fit to neural data our CNNs model the transformations from predictors to neural activity (or from neural activity to action). We therefore set out to quantify whether these networks implement a linear or nonlinear function as a proxy to classifying the actual transformations between stimuli, behavior and neural activity.

To measure nonlinearity encapsulated by the CNN we calculated two metrics that quantify different aspects of nonlinear computation (Figure 3A and S3A). Linear functions are characterized by a constant first derivative and the absence of curvature. Hence, our first metric uses the *Hessian* (matrix of second-order derivatives) of the network output with respect to the input at various time points to quantify the curvature of the function approximated by the CNN (see Methods). The second metric is the "nonlinearity coefficient" (NLC) which measures the deviation of the CNN from its linear approximation by comparing the variance of true network outputs to that of a linear approximation[33,34].

We tested these metrics on ground-truth data in which we transformed a predictor to varying degrees with either a linear or a nonlinear function. We then trained the CNN to learn the transformation between the predictor and the mixed output and subsequently calculated the curvature and NLC metrics. Increasing the contribution of an arbitrary nonlinear function (i.e., $\tanh^2(x^3)$) leads to gradual increases in both nonlinearity metrics (Figure 3B). Importantly, these metrics do not simply quantify a loss of linear correlation. Calculating the derivative of a function is a linear operation, and increasing the contribution of a derivative of the predictor indeed does not increase either the curvature or the NLC metric in spite of reducing linear correlation between the predictor and the response (Figure 3C and S3B-D). In some cases, the performance of the metrics differ. When increasing the contribution of a rectified version of the predictor, the curvature metric correctly tracks this nonlinearity while the NLC does not (Figure 3D). We therefore systematically evaluated the classification performance of the two non-linearity metrics. To this end we generated random linear and non-linear transformations of the predictors (see Methods), trained the CNN and calculated the nonlinearity metrics. ROC analysis shows that both metrics consistently rank a nonlinear transformation higher than a linear one (in 86 % of cases the NLC will rank the nonlinear transformation in a random pair higher while the curvature metric will do the same in 92 % of cases, Figure 3E). Training a logistic regression model to combine the two metrics and subsequently evaluating its performance on test data further increased the classification performance. Importantly, this model greatly improved the true-positive-rate for thresholds with a low false-positive-rate compared to the curvature metric; a feature that is highly desirable when applying the classification to real-world data (Figure 3E).

In summary, we can further increase the interpretability of our CNN model and classify the transformations the model encapsulates according to their nonlinearity. Since the CNN encapsulates the transformations that are enacted by neural circuits between stimuli and neural activity or neural activity and behavior, this information provides important insight into the computations that give rise to neural activity.

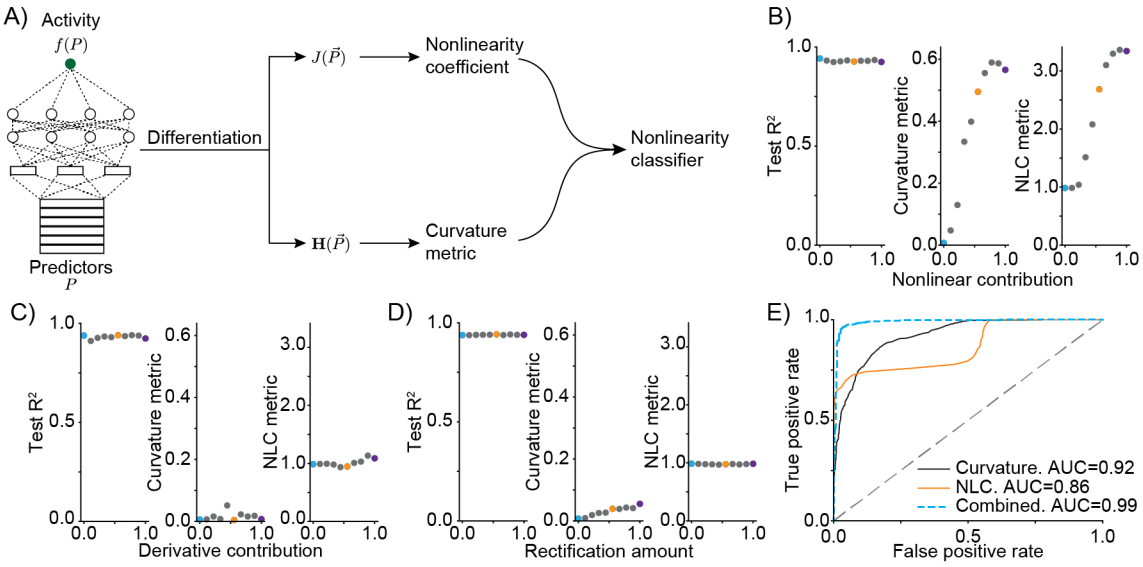

Figure 3: Measures of nonlinearity separate linear from non-linear predictor-activity relationships
A) Schematic of network decomposition and parameters used for the metrics. C is related to the curvature of the function which would be 0 for a linear network. The NLC on the other hand quantifies the relationship between the variance induced by a linear approximation of the network around the data mean versus the true output variance.
B) Mixing varying degrees of a nonlinear response function with a linear response ("Nonlinear contribution") and its effect on network performance (left, fraction of explained variance on test data), the curvature metric (middle) and the NLC. Colored dots relate to the plots of linear correlations in Figure S3.
C) As in B) but mixing a linear function of a predictor with a linear transformation of the predictor, namely the first order derivative.
D) As in B) but mixing a linear function of a predictor with a rectified (nonlinear) version.
E) ROC plot, comparing classifier performance of nonlinearity for the curvature metric (black), the NLC (orange) and a logistic model combining both metrics (dashed blue line). Note that the combined model allows to identify a large fraction of true positives while minimizing false positives.

Characterization of receptive fields

Receptive fields describe the stimulus features driving a neuron's response[35]. They can reveal integration times (those times over which coefficients of the receptive field are different from zero) and zero-crossings within the receptive field indicate that a neuron encodes the derivative of a stimulus across time (e.g., velocity encoding neurons) or that it detects edges across space. A powerful approach to characterize receptive fields of neurons is spike-triggered analysis[36–39]. In this approach white-noise stimuli are presented to an animal and the stimulus history before each spike is analyzed. Averaging the stimuli leading up to a spike yields the spike-triggered-average (STA). The STA characterizes those stimulus features that increase the average firing rate of the neuron, classically referred to as its receptive field. Eigenvectors of the spike-triggered covariance (STC) matrix on the other hand reveal receptive fields that change the variance of the firing rate rather than the average. Together, STA and STC therefore characterize how key features of the stimulus drive the neural response: The projection of the stimulus along the STA influences the average firing rate, while the projections along different components of the STC alter the response variance.

The first two terms of the Taylor expansion of the network fit by MINE suggest a correspondence to both STA and STC. In the first order term of the expansion, the Jacobian has the same effect as the STA: it projects the stimulus and increases the average response. Eigenvectors of the Hessian on the other hand, mark directions that alter the variance of the response without affecting the average much like the components of the STC (see Methods). A key difference however is that spike triggered analysis is agnostic to nonlinearities associated with the receptive fields defined by the STA and

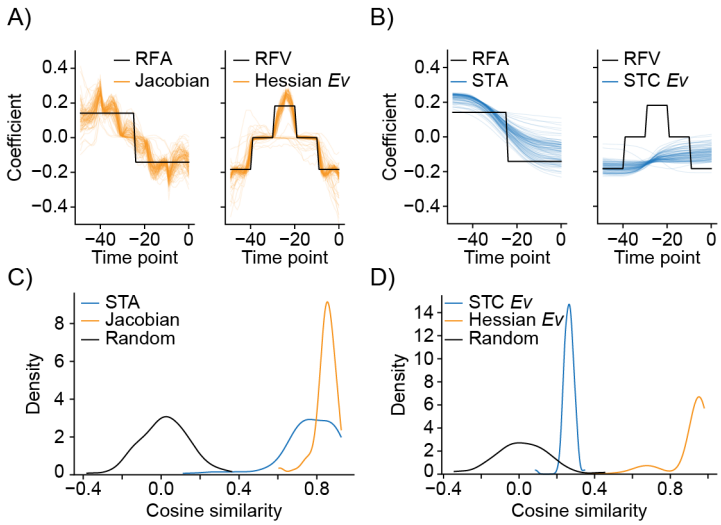

Figure 4: Comparison of MINE and spike-triggered analysis
A) Approximations of the receptive field affecting the average spike rate (RFA) using the *Jacobian* of the CNN fit by MINE (left) and of the receptive field affecting the variance of the spike rate (RFV) using the eigenvector of the *Hessian* of the CNN fit by Mine. Black lines are the true receptive fields, thin orange lines are derived approximations across $N = 100$ randomly generated, smoothly varying stimuli.
B) Same as A) but approximations derived using spike-triggered analysis. The spike-triggered average (STA) left and eigenvector of the spike-triggered covariance matrix (STC) right are compared to the RFA and RFV respectively.
C) Across $N = 100$ randomly generated, smoothly varying stimuli, distribution of cosine similarities between the RFA and the MINE *Jacobian* (orange), the spike-triggered average (blue) and a random vector (black).
D) As in C) but distribution of cosine similarities between the RFC and the MINE *Hessian* eigenvector (orange), the eigenvector of the spike-triggered covariance matrix (blue) and a random vector (black).

STC. For MINE on the other hand, nonlinear effects are largely contained in higher-order terms of the Taylor expansion.

We therefore wanted to test to what extent MINE could identify receptive fields that arise from the model underlying spike triggered analysis. To this end we generated a white-noise stimulus (Figure S4A) and created a simulated spike rate which depends on two receptive fields: One, which affects the average response (RF_A) and one which only changes its variance (RF_V) (Methods). We subsequently analyzed the relationship of the stimulus and the spike rate using MINE, fitting the network and extracting both the Jacobian as well as the eigenvectors of the Hessian. Indeed across many randomly generated stimuli, the Jacobian matched the receptive field increasing the average response, RF_A (Figure S4B and D). The eigenvectors of the Hessian with the largest two eigenvalues on the other hand always contained the receptive field increasing the response variance, RF_V (Figure S4B and E). As expected, performing actual spike-triggered analysis on this data recovered the same receptive fields through STA and STC (Figure S4C-E).

Presenting white-noise stimuli is generally not a practical solution in in-vivo experiments. Stimuli experienced by animals are rarely white noise and behaviors cannot be drawn from a distribution by the experimenter. We therefore repeated the analysis using a stimulus that varies smoothly across time, mimicking ongoing stimuli or behaviors (Figure S4F). Again, fitting MINE on the data, the expected receptive fields RF_A and RF_V were recovered but the correspondence was slightly worse compared to using white-noise stimuli (Figure 4A and C-D). This makes sense given the strong autocorrelations in the stimulus. Analyzing the same data with spike-triggered analysis led to considerably worse correspondences, especially for RFV (Figure 4B and C-D). While whitening the stimuli could ame-

liorate this effect for the STA fully recovering RF_A, the STC analysis failed to recover RF_V regardless (Figure S4G-H). This indicates that MINE is better in recovering receptive fields for smoothly varying inputs, leading to less degradation than spike triggered analysis.

Overall these results suggest that MINE can recover the types of receptive fields of neurons that can be obtained with spike triggered analysis under a broader set of biologically relevant stimulus conditions.

MINE characterizes cortical sensori-motor processing

Encouraged by MINE's ability to identify responses related to behavior and stimuli and its ability to characterize the nature of that relationship, we wanted to test the method on biological data. We applied MINE to a publicly available wide-field calcium imaging dataset recorded in the mouse cortex[40]. The dataset consists of 22 task related predictors (stimuli, reward states, instructed and non-instructed movements) and calcium activity time series across 200 temporal components which were used to compress each session's widefield imaging data[7] from 13 mice. It had previously been analyzed using linear regression[7]. We analyzed one session from each of the 13 mice present in the dataset with MINE (Figure 5A). As in the ground-truth data, we split the data into a training set (first two thirds of the session time) and a test-set (last third). To be able to capture not only activity caused by past stimuli but also preparatory activity leading up to movements, we shifted the predictor traces with respect to the activity trace (see Methods). As expected, correlations between MINE predictions and true activity on the test dataset is overall smaller than on the training set; however, in the majority of cases the CNN generalizes well to the new data (Figure 5B). Given that many individual neurons contribute to each pixel and temporal component within this dataset, and that a lot of brain activity is likely unrelated to the behavioral task at hand, we chose a lenient correlation cut-off of $r = 0.1$ for the test-data to decide that a component had been successfully fit by MINE (Figure 5B-C). On average this led to the identification of more than 91 % of all components per session (Figure 5C) on which we computed the Taylor metric and nonlinearity probability . Notably, MINE assigns very low probabilities of nonlinearity to all but five components. This means that the relationships between predictors and neural activity are linear (Figure 5D). While this may seem surprising given the high nonlinearity of cortical processing, it is likely caused by the low resolution of the data. Each component blends the activity of hundreds of thousands of neurons. Averaging across many individual neurons which each may have their own nonlinear responses likely obfuscates nonlinear effects.

Across all sessions we found a broad representation of predictors(Figure 5E). As expected from the findings of Musall et al.[7], uninstructed movements appear overrepresented sporting the largest Taylor metric in more than half of the fit components. We next mapped the results of MINE back into acquisition space and recalculated the Taylor metrics for each imaging pixel for one example session. The obtained results further validate MINE's utility on biological data. Specifically, visual stimuli are shown to contribute most strongly to responses in visual cortical areas with the expected left-right asymmetry (Figure 5F). At the same time, comparing visual sensory responses to whisk responses shows that cortical regions enhanced for whisking (Figure 5G) correspond very well to regions marked strongly with a whisk event kernel in Figure 2 of Musall et al.[7]. Furthermore, instructed left and right grab events largely contribute to motor cortical regions, again with the expected left-right asymmetry (Figure 5H). Repeating this procedure on all sessions (Figure S5) reveals general agreement. But we note that not all sessions seem to have regions that are as clearly related to the chosen inputs according to Taylor analysis which is especially apparent for left and right instructed grabs (Figure S5C).

We next sought to determine the receptive fields that govern stimulus processing in individual pixels[18,35]. We extracted the receptive fields across pixels that are strongly related to either left/right visual stimuli or

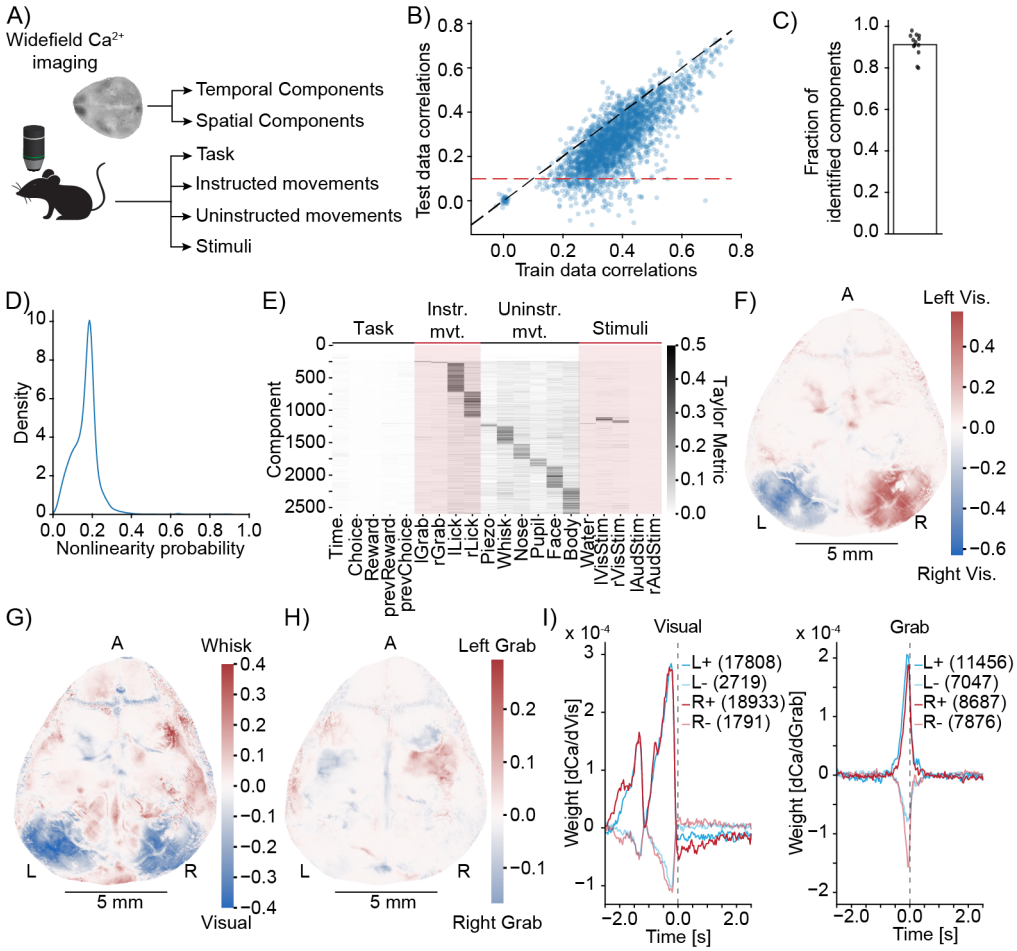

Figure 5: MINE identifies cortical features of sensori-motor processing during a learned task

A) Simplified schematic of the widefield imaging experiment.

B) MINE test data vs. training data correlations on 200 temporal components from 13 sessions across 13 mice. Black dashed line is identity, red dashed line is the test correlation threshold to decide that a component had been identified by MINE.

C) In each of the 13 sessions, the fraction of identified components (dots) as well as the average (bar).

D) Across 200 components each in the 13 sessions the distribution of nonlinearity probabilities computed by MINE.

E) Across all components from all 13 sessions that have been identified by MINE the Taylor metrics that were significantly larger than 0. Components (rows) have been sorted according to the predictor with the maximal Taylor metric.

F) Per pixel Taylor-metric scores for the right visual stimulus ("rVisStim") subtracted from those of the left visual stimulus ("lVisStim"). A=anterior, L=left, R=right.

G) As in F) but the sum of the visual stimulus Taylor-metrics ("lVisStim+rVisStim") has been subtracted from the Taylor-metric of the whisking predictor ("Whisk").

H) As in F) but the Taylor-metric for the right grab predictor ("rGrab") has been subtracted from the Taylor-metric of the left grab predictor ("lGrab").

I) Clustering of per-pixel receptive fields, separating excitatory (bold lines) from inhibitory responses (pale lines). Pixels were selected to only include the top 10% Taylor metrics for visual (left plot) and grab (right plot) predictors; left blue, right red. Numbers in parentheses indicate cluster size. The gray dashed line indicates time 0, i.e. the time at which calcium activity is measured. Note that the sensory receptive field (visual) is exclusively in the past (future stimuli do not drive the neurons) while the motor receptive fields (grap) slightly overlap with the future, indicating that neurons ramp up activity before the behavioral action.

left/right instructed grabs. We broadly clustered the receptive fields into two groups (see Methods) to separate excitatory and inhibitory effects. Receptive fields for left and right visual stimuli in contralateral brain regions are highly similar to each other (Figure 5I, left). The excitatory effect is stronger than the inhibitory effect (larger coefficients in the positive receptive fields) and both are clearly biphasic, indicating that events around the time

of the stimulus as well as $\sim 1.5\,\mathrm{s}$ in the past, strongly influence the activity of the visual neurons. We note that this biphasic structure mimics the linear regression kernel in Figure 3b of Musall et al.[7]. The visual receptive fields have essentially zero weight after the current time which is expected since future stimuli are unlikely to influence neural activity. The receptive fields of left and right grab neurons on the other hand are much sharper, indicating that these neurons influence movement over short timescales (Figure 5I, right). Furthermore, the grab-related receptive fields contain coefficients different from baseline for up to 100 ms into the future. This suggests preparatory activity, i.e., that future movements are reflected in current neural activity.

In summary, the results presented above demonstrate the applicability of our method to biological data and the potential for identifying diverse feature sets of sensorimotor processing.

Functional characterization of thermoregulatory circuits

Since MINE was able to extract expected biological features from a mouse cortical dataset, we sought to use it to gain novel insight into zebrafish thermoregulatory circuits. Temporal transformations of stimuli are a notable feature of how zebrafish process temperature stimuli. In previous research we identified the neurons and centers in the larval zebrafish brain that process temperature: specifically, neurons that compute the rate of change of the temperature stimulus[41] as well as neurons whose activity is consistent with integration of temperature fluctuations[42]. Due to the nature of these transformations, which were unknown *a priori*, regression-based approaches failed to identify neurons involved in temperature processing. We therefore previously used clustering to identify these neurons. However, because behavior is stochastic, one particular drawback of clustering was that it precluded identifying neurons that integrate thermosensory stimuli and behavioral actions. Since MINE can learn and capture both temporal transformations and interactions, we set out to gain deeper insight into thermoregulatory processing.

To classify neurons by clustering either requires presenting the same stimulus to every animal and neuron while imaging or it requires a strategy of clustering event-triggered activity. With MINE we do not need to enforce any stimulus constraint. As a result we were able to probe a wider variety of stimulus conditions. We imaged a total of 750 planes across 25 larval zebrafish that expressed the nuclear calcium indicator H2B:GCaMP6s[43] in all neurons and the excitatory neuron marker vglut2a-mCherry[44] in all glutamatergic neurons. This dataset provided slightly more than fourfold coverage of the larval zebrafish brain. On each imaging plane we presented a heat stimulus generated by randomly composing sine-waves with frequencies between 0.005 Hz and 0.075 Hz (Figure 6A and S6A-B). We concurrently recorded elicited tail motion at 250 Hz. We segmented the imaging data using CaImAn[45] which identified $\sim 433,000$ active neurons across our dataset. From the tail motion data, we extracted (a) swim starts and (b) tail features that correlate with swim displacement ("Vigor") and turn angle ("Direction") in free swimming experiments (Figure 6A and S6C).

We used the information about stimulus and elicited behaviors across time as inputs to MINE fitting CNNs to each neural response (Figure 6B and C) using the initial two thirds of time as training data and the last third as test data. Notably, due to the random nature of both our stimulus and the elicited behavior, test and training data were largely uncorrelated (Figure S6B). Predictive power over the test data is therefore a good indication that the neural network model generalizes and truly captures the relationship between stimulus and behavioral data and neural activity. We chose 50 % of explained variance on the test data, corresponding to a correlation of $r = \sqrt{0.5}$, as our cutoff to determine whether a neuron can be modeled by MINE. Using cyclic permutations of our dataset as a control revealed that this cut-off corresponds to a 93-fold enrichment over the control data (Figure S6D).

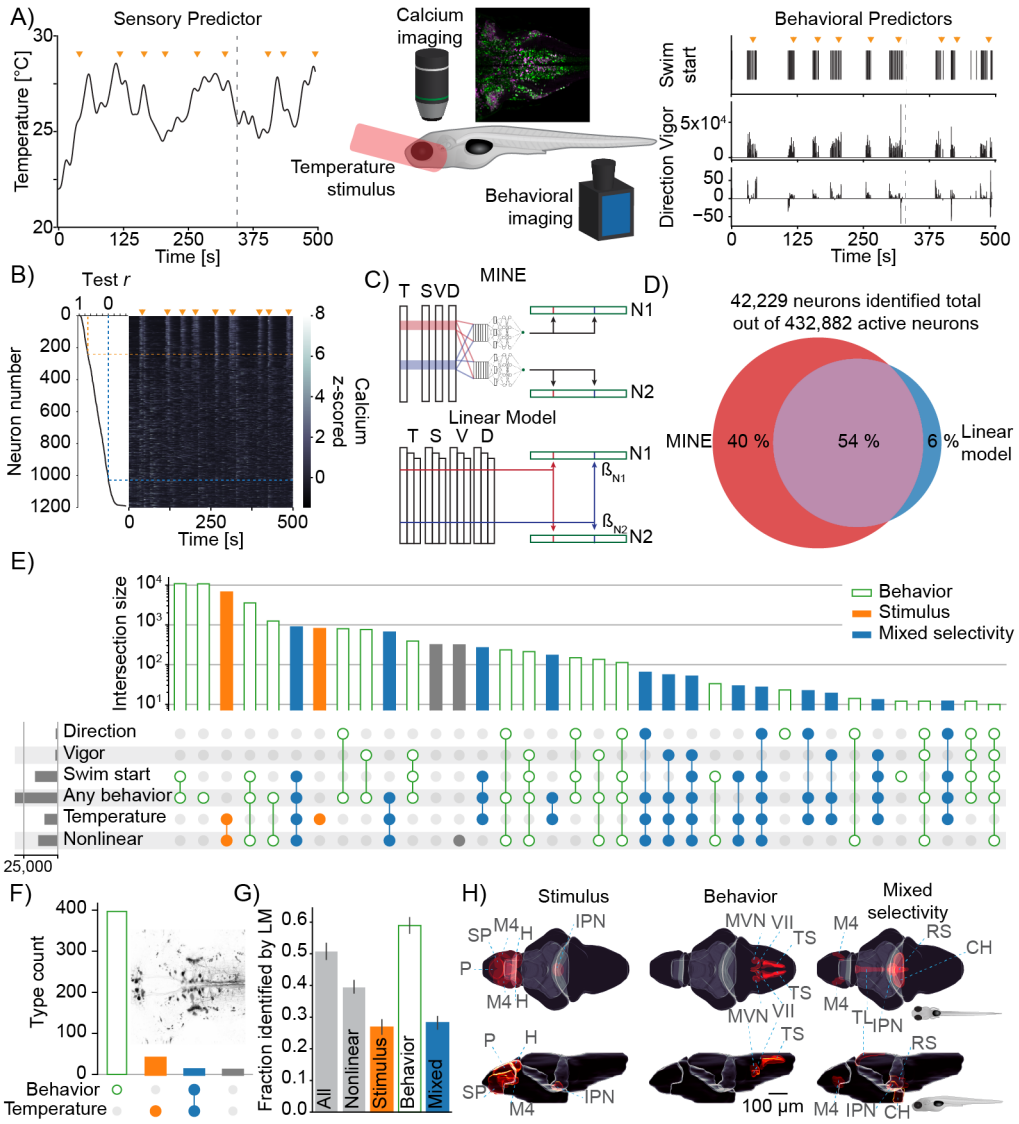

Figure 6: Using MINE to probe larval zebrafish thermoregulatory circuits

A) Experimental design. Larval zebrafish expressing nuclear GCaMP6s in all neurons and mCherry in glutamatergic neurons are imaged under a 2-photon microscope while random heat stimuli are provided with a laser and behavior is inferred through tail motion, middle. Left, example temperature trajectory during one trial. Right, behavioral responses recorded during the same trial.

B) Deconvolved calcium traces of all neurons identified in the plane imaged during the same trial as shown in A) (heatmap), sorted by the test correlation achieved after ANN fitting (plot on the left). Orange arrowheads mark the same time-points in A) and B). Orange dashed line indicates the fit cutoff used for deciding that a neuron was identified by MINE, blue line marks pearson correlation of 0.

C) Illustration comparing MINE to the comparison linear regression model which uses time-shifted predictors to have access to the same information as the convolutional neural network used in MINE.

D) Venn diagram illustrating what fraction of CaImAn identified neurons across the whole dataset were identified by MINE, the comparison LM model or both.

E) Upset plot of functional classes identified by Taylor analysis across 25 fish. Bar-plot at the top indicates the total number of neurons in each class on a logarithmic scale. Dot-plot at the bottom marks the significant Taylor components identified in each functional class. Functional classes are sorted by size in descending order. Horizontal barcode plot on the right, indicates the total number of neurons whose activity depends on the given predictor. Orange filled bars mark classes only driven by the stimulus, green open bars those only driven by behavioral predictors while blue bars mark classes of mixed sensori-motor selectivity.

F) Classification of neurons labeled by reticulospinal backfills (inset shows example labeling) across six fish. Bar color as in E).

G) For different functional neuron classes identified by MINE the fraction also identified by the linear comparison model.

H) Maps of top five Z-brain regions enriched for Stimulus related neurons (left), Behavior related neurons (middle) and neurons of mixed selectivity (right). Dorsal view top, lateral view bottom. Enriched regions are marked in red. CH: Caudal Hypothalamus; H: Habenula; IPN: Interpeduncular nucleus; M4: Telencephalic migrated area 4; MVN: Medial Vestibular Nucleus; P: Pallium; RS: Raphe Superior; SP: Subpallium; TL: Torus Longitudinalis; TS: Rhombencephalic Vglut2 Stripe 2 and Vmat2 Stripe 2 and Glyt2 Stripe 2; VII: Facial motor and octavolateralis efferent neurons.

As a comparison, we fit a linear regression model to this dataset. The inputs included time-shifted versions of stimulus and behavioral variables to allow the model to learn temporal transformations[7] and thereby put it on the same footing as the CNN model (Figure 6C). We used ridge-regression[9] to improve generalization of the linear model. At our designated cut-off a total of $\sim 42,000$ neurons were identified using either MINE or regression. Importantly 40 % of these neurons were exclusively identified by MINE (Figure 6D), indicating the superiority of the model-free approach. In fact, MINE consistently identifies more neurons regardless of the cut-off imposed on test data (Figure S6E).

Using the Taylor decomposition approach described earlier, we classified the neurons according to which stimulus and behavioral features drove their activity. We chose a stringent cut-off, requiring the Taylor metric to be significantly above 0.1 with a p-value < 0.05 on the whole dataset (Bonferroni correction; effective p-value $\sim 1.26x10^{-6}$), to determine if a predictor influenced the activity of a neuron. Additionally, we applied our nonlinearity model for classifying neurons (deemed non-linear whenever the probability of nonlinearity was greater than 0.5) (Figure S6F-H). This led to the identification of 34 functional neuron classes (Figure 6E). We ignored nonlinear interaction terms since they had very small Taylor metrics. As a control, we labeled reticulospinal neurons in six fish via spinal backfills and as expected the vast majority of these neurons were classified by MINE to be driven by behavioral features (Figure 6F). Across the brain, we identified multiple functional classes of neurons with mixed selectivity, i.e. neurons with activity jointly driven by the temperature stimulus and behavioral outputs (Figure 6E, blue filled bars). These mark a previously unidentified group of neurons in the thermoregulatory circuit which might play a key role in behavioral thermoregulation by giving the animal the ability to relate behavioral output to temperature changes and thereby characterize the thermal landscape. Analyzing the success of the linear comparison model according to functional class revealed a bias towards the identification of behavior related activity while stimulus or mixed selectivity neurons are underrepresented in the pool identified by the linear model (Figure 6G). This points to potential problems of bias when identifying neurons by means of linear models.

We registered all our imaging data to a standard zebrafish reference brain (Z-Brain[46]). This allowed us to assess the distribution of identified functional neuron types throughout the zebrafish brain (Figure 6H and S6I-L). While Stimulus-, Behavior- and Mixed-selectivity neurons are generally broadly distributed, they are enriched in specific brain regions (Figure 6H and S6J-L). We found stimulus driven neurons to be enriched in telencephalic regions, as well as the habenula and its output region the interpeduncular nucleus. As expected, behavior driven neurons are enriched in hindbrain regions. Mixed selectivity neurons occupy some regions shared with stimulus driven neurons such as telencephalic area M4 and the interpeduncular nucleus but are also strongly enriched in the raphe superior as well as the caudal hypothalamus and the torus longitudinalis (Figure 6H). Our current dataset did not cover sensory ganglia well (especially not the trigeminal ganglion) with the exception of the olfactory epithelium where we found temperature sensitive neurons in a medial zone (Figure S6I). This is in line with reports in mice and frogs that describe specific thermosensitive neurons in the olfactory epithelium of these species[47–49].

Overall these results demonstrate that functional cell types identified by MINE segregate spatially within the brain, likely forming organizational units within the thermoregulatory circuit.

Mapping computational features to the zebrafish brain

We next sought to localize computational features of thermoregulatory circuits in the brain by subdividing neural classes according to the behavioral features they control and the sensory features they extract. Analyzing brain regions for enrichment of encoding different behavioral features suggests a segregation in the con-

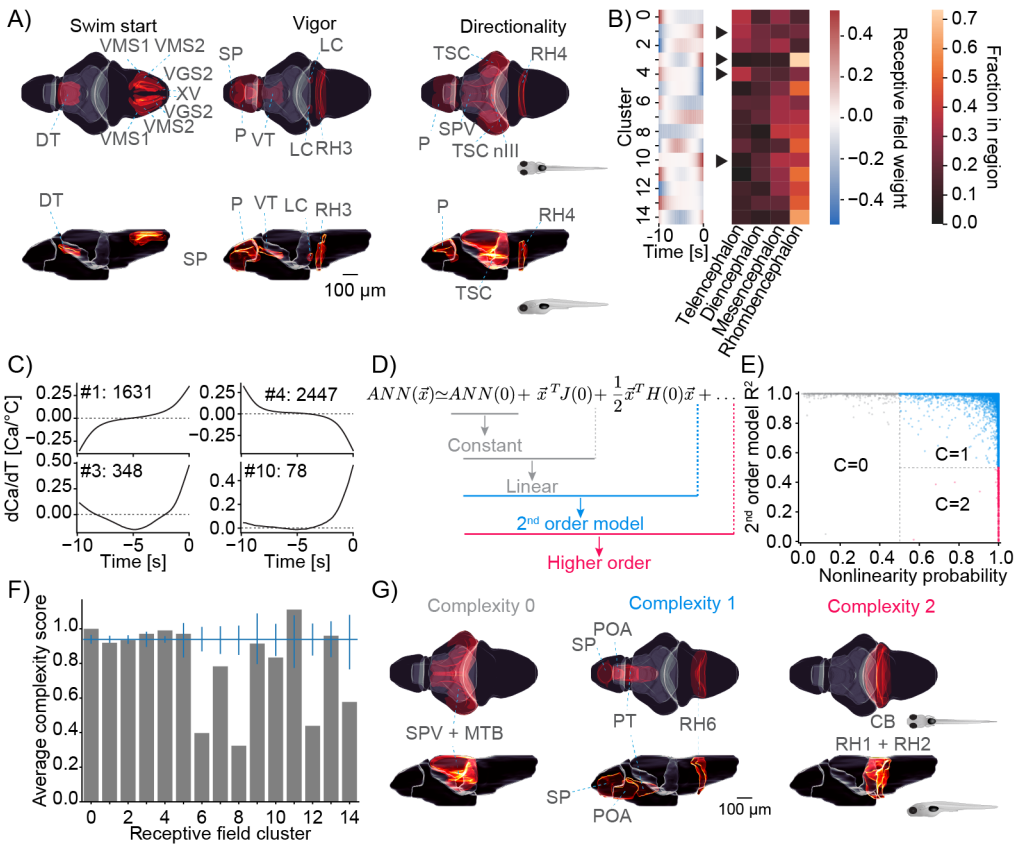

Figure 7: Functional subdivisions of thermoregulatory circuits

A) Top five brain regions enriched for neurons that encode specific aspects of movements. Swim starts left, vigor (related to swim displacement) middle and directionality (related to turn angle) right. DT: Dorsal Thalamus; LC: Locus Coeruleus; nIII: Tegmental Oculomotor nucleus III; P: Pallium; RH3: Rhombomere 3; RH4: Rhombomere 4; SP: Subpallium; SPV: Tectum Stratum Periventriculare; TSC: Torus semicircularis; VGS2: Vglut2 Stripe 2; VMS1: Vmat Stripe 1; VMS2: Vmat Stripe 2; VT: Ventral Thalamus; XV: tenth nerve vagus motorneuron cluster.

B) Sub-clustering of stimulus-selective neurons according to their temporal receptive field (left heatmap). Right heatmap visualizes for each cluster what fraction of that cluster is present in the major four subdivisions of the zebrafish brain. Arrowheads indicate differentially distributed example clusters highlighted in C).

C) Temporal receptive fields of example clusters. Each plot shows the influence of a change in temperature on the activity (as measured by Calcium) of a neuron within the cluster. Numbers reflect the number of neurons present in each given cluster. Dashed lines indicate 0 where the number of 0-crossings of each receptive field indicate if the neuron responds to absolute temperature value (no crossings, cluster 10), to the first derivative (velocity of temperature, increasing, cluster 1; decreasing cluster 4) or to the second derivative (acceleration of temperature, cluster 3).

D) Schematic of how the Taylor expansion of the fitted ANN around the mean of the data indicates what order terms (constant, linear, 2nd or higher-order) are used in the transformation of the stimulus to the observed neural activity.

E) Scatter plot of the goodness-of-fit of a second order Taylor expansion model against the nonlinearity probability. These two metrics together divide the neurons into three classes of increasing complexity.

F) Average complexity of each receptive field cluster shown in B) (gray bars). Blue horizontal line reveals the total average complexity and vertical blue lines indicate bootstrapped 95% confidence intervals around the average complexity. If the gray bar is above or below that interval, the complexity within that cluster deviates significantly from the data average complexity.

G) For each complexity class the significantly enriched brain regions (2 regions for complexity 0, 4 for complexity 1 and 3 for complexity 2). CB: Cerebellum; MTB: Medial tectal band; POA: Preoptic area of the hypothalamus; PT: Posterior tuberculum; RH1: Rhombomere 1; RH2: Rhombomere 2; RH6: Rhombomere 6; SP: Subpallium; SPV: Tectum Stratum Periventriculare.

trol of swim-starts, swim speed (Vigor) and turn angle (Directionality). Even though behavior related neurons are enriched in the hindbrain compared to sensory or mixed-selectivity neurons (Figure 6H), subdividing the neurons reveals a broader distribution (Figure 7A; Figure S7A-C). Swim start neurons are enriched in the dorsal thalamus as well as the medial hindbrain, while the ventral thalamus and some forebrain regions as well as

the locus coeruleus and anterior hindbrain preferentially encode swim speed. Forebrain regions together with the anterior hindbrain and the torus semicircularis on the other hand are enriched for neurons encoding turning (Figure 7A). Notably, these neurons could be involved in controlling these behavioral features or alternatively maintain copies of the motor commands for other purposes.

To subdivide stimulus driven neurons, we performed clustering on their receptive fields which we derived from the fit CNN as described above (Figure 7B-C; Figure S7D). A clustering approach based on the cosine similarity of receptive fields (see Methods) resulted in 15 clusters. Each cluster represents a functional neuron class that extracts specific features of the temperature stimulus (Figure 7B, left). Localizing different clusters in the brain reveals an anatomical organization of processing. Clusters one and four are enriched in the forebrain (Figure 7B, right) and their receptive fields suggest that they are most responsive to heating and cooling stimuli over slow timescales, respectively (Figure 7C), as evidenced by the zero-crossing of their receptive fields. Cluster three on the other hand computes the rate of heating over short timescales (Figure 7C) and is almost exclusively localized to the hindbrain (Figure 7B, right). Cluster 10 represents neurons that are excited by warmer temperatures since the receptive field almost exclusively has positive coefficients (Figure 7C). This neuron type is found predominantly in hind- and midbrain regions (Figure 7B, right).

Lastly, we subdivided neurons according to computational features themselves, i.e., those that occur between the sensory input and the observed activity of the neurons. Specifically, we were interested in the complexity of these computations (Figure 7D). MINE computes the probability that the relationship of predictors and the neural activity is nonlinear and temperature responsive neurons can be grouped into those that are linear and those that are nonlinear (Figure 6E). We note that these are not necessarily absolute classifications, meaning that we could transform the stimulus predictor such that some nonlinear relationships would become linear. Instead, this classification points to differential processing upstream of the observed neural activity in linear vs. nonlinear neurons. We used the Taylor decomposition of networks to identify contributing predictors (see above). The decomposition also provides a window into the complexity of nonlinear neurons. In particular, we asked how well a Taylor series around the data average truncated after the 2nd order term could approximate the true neural activity. If this approximation is good, the neural activity could be described by a second-order regression model (Figure 7D). If, however, the Taylor series fails to explain most of the variance in the neural activity we would conclude that the true relationship between sensory inputs and neural activity requires higher order terms and hence a more complex model (Figure 7D). The goodness-of-fit of the 2nd order Taylor series therefore allows us to further divide neurons classified as nonlinear. Plotting the goodness-of-fit of the 2nd order model against the nonlinearity probability shows that all except one neuron considered linear (nonlinearity probability < 0.5) are also well explained by the 2nd order model (Figure 7E). This serves as a sanity check for our approach. Among nonlinear neurons we used a cut-off of 50 % explained variance by the 2nd order model to divide them into two complexity classes (Figure 7E). Importantly, complexity is mapped onto both functional (Figure 7F) and anatomical features (Figure 7G; Figure S7E-G) indicating that this division is not arbitrary but rather indicates a meaningful difference in neuron function. Specifically, averaging the complexity score for different clusters according to receptive field, reveals that four of these clusters have much lower complexity than expected indicating that most neurons in these clusters have responses that linearly depend on the temperature stimulus (Figure 7F). Anatomically, nonlinear neurons of complexity one are enriched in ventral forebrain regions while neurons of complexity class two are found in the cerebellum and anterior hindbrain (Figure 7G). This is in line with our previous observation that neurons in the cerebellum respond to increased temper-

class MineData

- *correlations_trained* (n_neurons x 1)
  Correlation of CNN prediction and activity on training portion of data.
- *correlations_test* (n_neurons x 1)
  Correlation of CNN prediction and activity on test portion of data.
- *taylor_scores* (n_neurons x n_components x 2)
  Taylor metric for each component (predictor and first order interaction terms). The first entry along the last dimension is the mean score, the second entry is the bootstrap standard error.
- *nl_probabilities* (n_neurons x 1)
  Nonlinearity probability according to model.
- *mean_exp_scores* (n_neurons x 1)
  Goodness of fit of 2nd order taylor model around the data mean to derive complexity.
- *jacobians* (n_fit_neurons x (n_timepoints x n_predictors))
  For each fit neuron the receptive field of each predictor across time.

- *save_to_hdf5(file_object, overwrite=False)*
  Saves the result data to an hdf5 file or group.

class Mine

- *train_fraction* (float)
  Which fraction of the data is used for training with remainder used for testing.
- *model_history* (integer)
  Number of timepoints the model receives as input.
- *corr_cut* (float)
  If test correlation is less than this value for a neuron, it is considered "not fit".
- *compute_taylor* (bool)
  If true, compute taylor metrics and nonlinearity probablility..
- *compute complexity* (bool)
  If true, compute mean expansion score.
- *return_jacobians*(bool)
  If true, return receptive fields.
- *taylor_look_ahead* (integer)
  The number of timepoints to predict ahead when calculating taylor metrics.
- *taylor_pred_every* (integer)
  Every how many frames a taylor expansion should be performed to calculate taylor metrics.

- *analyze_data(pred_data: List, response_data: Matrix) -> MineData*
  Takes a list of n_timepoint long predictors and a matrix of n_neurons x n_timepoints size and applies MINE iterativel to fit and characterize a CNN relating all predictors to each individual neuron.

Example Usage:

```
predictors = [Stimulus, Behavior, State]
responses = ca_data
# NOTE: If predictors and response aren't z-scored
# (mean=0; standard deviation = 1) Mine will print
# a warning
miner = Mine(2/3, 50, 0.71, True, True, False, 25, 5)
result_data = miner.analyze_data(predictors, responses)
all_fit = result_data.correlations_test >= 0.71
is_nonlinear = result_data.nl_probabilities > 0.5
is_stim_driven = (result_data.taylor_scores[:, 0, 0] - 3 x result_data.taylor_scores[:, 0, 1]) > 0
```

Figure 8: The programmatic interface to MINE

ature but do not encode the amplitude of the stimulus, indicating strong rectification[41].

In summary these data reveal the power of MINE to identify important computational features leading to a detailed classification of neurons into functional types. This automated, unbiased and model-free analysis has the potential to greatly aid the analysis of large-scale neural data. To facilitate the adoption of MINE we expose its functionality through a simple Python interface. This interface allows fitting the corresponding CNN to neural data and extracting all metrics (Figure 8).

Discussion

A common goal in analyzing neural data is to relate the activity of neurons to sensory stimuli, behavioral actions, internal states or other features observed during ongoing behavior. To explore this connection, we neuroscientists often define a model dictating the structure of this relationship. When well thought-out, these defined models have the great advantage that their parameters can be interpreted in a biologically meaningful manner. However even flexible models, if defined *a priori*, run the risk of not being able to match transformations occurring in the less than intuitive biological brain. Model-selection based on probabilistic programs overcomes some of these challenges, allowing for a model-free and flexible approach while maintaining predictive power[50]. One drawback of this approach however is reduced interpretability of the nature of the relationship between inputs (in our case stimuli, behaviors, etc.) and the output (in our case neural activity). Here we have presented a method, "Model-free Identification of Neural Encoding (MINE)," that uses simple convolutional neural networks to relate behaviors, stimuli or internal states to concurrently recorded neural activity. Despite

the method's increased flexibility over predefined models, MINE maintains interpretability by applying Taylor decomposition and concepts from explainable machine learning. As an example, MINE identified more neurons related to the experimental task in the zebrafish brain with less bias than a regression model. In particular, MINE's ability to jointly model the relationship of neural activity to ongoing stimuli and stochastic behavior allowed identifying a new class of thermoregulatory neurons. These integrate temperature stimuli with information about behavioral actions. Moreover we quantified the computational complexity of neurons and characterized their receptive fields.

Different paradigms are used to interpret neural activity. MINE specifically supports an encoding view, inspecting which information is encoded by the neurons of interest. Artificial neural networks have however also been used from a decoding perspective, probing what information can be gleaned about ongoing behavior from neural activity[51,52]. Here we restricted the question of information encoding to individual neurons. Nonetheless, MINE could easily be extended to model joint population encoding. The most straight-forward approach to this problem would be to decompose the population activity through methods such as principal or independent component analysis. These components would then be fit by MINE instead of single neuron activity. Similarly, we only demonstrate the relationship of neural activity to external features (stimuli, behavioral actions) but neural population activity or local field potentials could be used as predictors.

Artificial neural networks have been used extensively as models of brains to identify principles underlying neural processing[53]. In the present work, convolutional neural networks exclusively serve as tools to find relationships between features of interest and the activity of individual neurons. The network is a stand-in for the circuits transforming these features upstream (for sensation) or downstream (for behavior) as well as the computational properties of the analyzed neuron. We are therefore interested in properties of the whole CNN instead of its individual units. Extracting the CNN properties aims to characterize the neuron of interest in terms of how it encodes the features. Such an approach has been used previously to characterize neural receptive fields, recognizing the power of ANNs in capturing nonlinear transformations. For example, characterizations of nonlinear visual receptive fields[18–21] and nonlinear auditory receptive fields[22]. We go beyond these approaches in this study. MINE handles and disambiguates multiple different inputs originating from different domains (such as sensory stimuli and behavioral actions). MINE furthermore performs a more detailed characterization producing information about the linearity of the relationship of inputs and neural activity. These advances are made possible through mathematical analysis of the network.

Different techniques aim at understanding how artificial neural networks arrive at their outputs given a set of inputs[54]. These techniques are generally referred to as "explainable machine learning" and are particularly important in understanding what features classifiers use to make their decisions. These include visualization techniques as well as layer-wise relevance propagation[55] which aims to identify important input features driving the output of a classifier. In our case, however, we were interested to know which inputs persistently (across experimental time) are involved in driving the output of the network, i.e. the activity of the modeled neuron. Notably, once the network is trained the different inputs are not separable anymore. This means that there is no guarantee that leaving out an unimportant input will not affect the output of the trained network. A workaround would be to train the network with subsets of inputs[25]. However, this becomes infeasible even with a moderate number of inputs since all combinations would need to be tested. We therefore chose a continuously differentiable activation function for our network layers. This enabled Taylor decomposition to transform the network, *point-by-point*, into a separable sum of terms depending on individual inputs. This approach allowed us to infer which specific inputs drive the network output and hence the

activity recorded in the modeled neuron. Notably this technique can also identify multiplicative interactions which indicate coincidence detection. Yet, while such interactions were evident in our ground-truth dataset, we found little evidence of multiplicative interactions in the biological data. This is likely due to added noise obfuscating the contribution of these terms which is comparatively small even on ground-truth data (Figure 2D). In fact, most information about a multiplicative interaction would be carried in the first derivative: namely the derivative with respect to one input would vary according to another input. We did not exploit this insight in the present work but it could be used to disambiguate additive integration from coincidence detection.

To define the linearity of the relationship between features and neural activity we used two complementary measures that allowed us to arrive at a well-performing classifier. On the one hand, we directly compute a measure of curvature of the function implemented by the network. On the other hand, we compute the nonlinearity coefficient, a concept from explainable machine learning that estimates the accuracy of a linear approximation of the network[33,34,56]. The artificial neural network fit by MINE must encapsulate the transformations between the chosen features and the neural activity in order to be able to predict the activity. The classification of linear versus nonlinear processing by the network should therefore apply to the neural transformations as well. We note however, that this is not an absolute metric. Comparing the linearity for neurons responding to the same feature (e.g., a stimulus) is useful, but making the comparison across different features likely is not. In our zebrafish dataset, most temperature driven neurons were classified as nonlinear while most behavior driven neurons were classified as linear. This however does not mean that the stimulus is processed less linearly than the behavior - it is likely a result of our stimulus feature being the raw temperature while our behavior features are removed from the nonlinear transformation that changes swim commands into appropriate tail motion.

Applying MINE to biological data revealed its potential to discover relationships between stimuli and behaviors and neural activity across species and imaging modalities. On a mouse cortical widefield imaging dataset MINE recovered previously published structure in the neural data. Task elements driving neural activity could be mapped to expected cortical regions and receptive field analysis revealed expected structure in visual receptive fields while the receptive fields associated with neurons encoding instructed movements showed preparatory activity. Performing 2-photon calcium imaging in larval zebrafish while simultaneously presenting random temperature stimuli and recording behavior, allowed a deeper dive into thermoregulatory circuits. MINE allowed us for the first time to identify neurons that integrate temperature stimuli and information about behavior. This was not possible previously since regression-based approaches failed to identify neurons encoding temperature while clustering to identify these neurons relied on removing stochastic motor-related activity[41]. Neurons integrating stimuli and behavior might serve an important role in thermoregulation. Specifically, they might allow larval zebrafish to estimate thermal gradients by comparing behavioral actions and their consequences in terms of changes in environmental temperature. Combining receptive field analysis with the idea of computational complexity, on the other hand, provides a way to functionally subdivide temperature responsive neurons in great detail. This will greatly aid future studies trying to understand thermoregulatory circuits. Notably, the functional features we identified map to anatomical features in the brain pointing to structural subdivisions mirroring function. The ability to compare neurons, even though we used randomized stimuli, indicates another big advantage of MINE. When recording neural activity during free exploration, across animals different neurons will be recorded under very different stimulus and behavior conditions. This means that their responses can usually be compared only by choosing different trigger events (certain behaviors, specific stimulus conditions) and analyzing neural activity around those. MINE on the other hand allows

comparing networks fit on the whole activity time series removing the requirement and bias of choosing specific triggering events.

One surprising finding in comparing the cortical and zebrafish brain responses was the prevalence of nonlinear processing in zebrafish and the apparent lack of it in the cortical dataset. This is likely due to the different spatial resolution in the datasets. While the zebrafish data has single neuron resolution this is not the case for the cortical wide-field data especially since we apply MINE not to individual pixels but activity components. Each of these components combines the activity of thousands of neurons. It is likely that this averaging obfuscates individual nonlinear effects.

We currently determine linear receptive fields[35,38] of fit neurons using the first order derivatives. Similar to using spike-triggered covariance to obtain non-linear relationships between stimuli and neural activity[57] one could analyze eigenvectors of the matrix of second order derivatives to get a deeper understanding of how this receptive field changes with stimuli in the case of neurons classified as nonlinear. Analysis on ground-truth data suggests that these eigenvectors indeed match results obtained through spike-triggered covariance analysis (Figure 4). One crucial advantage of MINE is that it does not rely on specifically designed stimuli unlike spike triggered analysis. For spike triggered analysis, stimuli should ideally be white noise (uncorrelated) and for covariance analysis intensities must be drawn from a Gaussian distribution[38,58]. These constraints are impractical, especially when the goal is to study neural activity in freely behaving animals. However, we note that in cases where at least the Gaussian distribution of stimuli can be maintained, adverse effects of correlations can theoretically be overcome[59]. Independent of MINE the structure of receptive fields is influenced by the acquisition modality. For our zebrafish data, we used a nuclear localized GCaMP-6s. This variant has a long decay time[43] and since MINE is fit on calcium data, effects of this decay time will appear in the extracted receptive fields. This likely explains why the receptive fields shown for the zebrafish data extend over many seconds. This issue could be overcome by better deconvolution of the imaging data which would require a higher acquisition framerate.

In summary, MINE allows for flexible analysis of the relationship between measured quantities such as stimuli, behaviors or internal states and neural activity. The flexibility comes from MINE being essentially model-free except for the choice of architecture and hyperparameters which limit the functions the network can possibly approximate. This advantage can be seen by MINE identifying more neurons than regression approaches on our zebrafish dataset (Figure 6D) and doing so in a less biased manner. The regression approach especially failed on stimulus driven units, leading to an overrepresentation of behavior-related activity (Figure 6G). The ability to learn arbitrary temporal transformations through the use of convolutional networks makes MINE particularly helpful in analyzing calcium data since MINE does not require convolving inputs with a "calcium kernel" that mimics temporal effects of the indicator[4]. Otherwise this kernel must be obtained through simultaneous electrophysiology recordings or has to be estimated by other means[4,60].

We propose that the fit convolutional neural network models the *computation* of upstream or downstream circuits of the neuron in question but we do not think or investigate if it models the *circuitry itself*. This could be investigated in the future but given the simplicity of the CNN used in MINE it is unlikely to provide interesting details in this manner. Bringing more advanced network-wide analysis methods to bear on the fit CNN could reveal other interesting features of the computation "surrounding" the neuron in question.

Methods

Animal handling and experimental procedures were approved by the Ohio State University Institutional Animal Care and Use Committee.

Code and data availability

All code used in this study is available in the following repositories:
MINE: https://github.com/haesemeyer/mine_pub
CaImAn preprocessing: https://github.com/haesemeyer/imaging_pipeline
Tail behavior processing: https://bitbucket.org/jcostabile/2p/
Raw experimental data has been deposited to DANDI:

- https://dandiarchive.org/dandiset/000235
- https://dandiarchive.org/dandiset/000236
- https://dandiarchive.org/dandiset/000237
- https://dandiarchive.org/dandiset/000238

Processed data has been deposited on Zenodo: https://zenodo.org/record/7017100. All data analysis was performed in Python; Tensorflow[61] was used as the deep learning platform.

Design and training of convolutional neural network

A network that was simple, and therefore quick to train, but maintained expressivity was sought for this study and a three-layer convolutional neural network was chosen. We note that the input to the network is a chunk of time equal to the size of the convolutional layers. A similar architecture could therefore be realized without separate convolutional layers. The first layer was built to contain 80 convolutional filters to generate enough pre-training variation to ease learning of temporal features important for the generation of the neural response. The convolutional layer was kept linear and followed by two 64-unit sized dense layers with swish activation function which subsequently fed into a linear output unit. The ground-truth dataset as well as part of a previous zebrafish dataset[41] were used to optimize the following hyperparameters: a sparsity constraint to aid generalization, the learning rate, the number of training epochs and the activation function. The activation function was to be continuously differentiable, a requirement for subsequent analyses. Dropout was added during training to further help with generalization, however the rate was not optimized and set at 50 %. The history length (10 second) of the network input was set somewhat arbitrarily but initial tests with shorter lengths led to problems of fitting the larval zebrafish dataset likely because of the slow calcium indicator time-constant of nuclear GCaMP 6s[43].
Related python file in the repository: model.py

Identifying contributing predictors by Taylor decomposition

To assess the contribution of individual predictors on the network output a metric based on the Taylor decomposition of the network was computed. The Taylor decomposition was truncated after the second derivative. For this study, Taylor decompositions including all terms for first- and second-order derivatives will be designated as

"full." The *Jacobian* (vector of first order derivatives of the output with respect to the input) and the *Hessian* (matrix of second order derivatives of the output with respect to the input) were computed using Tensorflow. For all analysis presented in the paper, Taylor expansions were performed every second (every five timepoints at our data-rate of 5 Hz) and predictions were made five seconds into the future (25 timepoints at our data-rate of 5 Hz).

The state of the predictors at time point t and the state of the predictors five seconds (in our case 25 samples) later was obtained. Subsequently these quantities, together with the *Jacobian* at time point t and the *Hessian* at time point t, were used to compute a prediction of the change in network output over five seconds. At the same time the true change in output was computed. Comparing the predicted changes and true changes via correlation resulted in an R^2 value. This value measures how well the full Taylor expansion approximates the change in network output across experimental time. We note that *predicted changes* in the output were compared via correlation rather than *predicted outputs* since we specifically wanted to understand which predictors are important in driving a change in output rather than identifying those that the network might use to set some baseline value of the output. In the following, $P(t)$ is a vector that contains the values of all predictors across all timepoints fed into the network (e.g., in our case 50 timepoints of history for each predictor) in order to model the activity at time t, $f(x)$ is the CNN applied to a specific predictor input, $J(x)$ is the *Jacobian* at that input and $H(x)$ the *Hessian*:

$$\vec{P} = P(t)$$

$$\vec{P'} = P(t + 25)$$

The Taylor expansion around $\vec{P}$:

$$\hat{f}(\vec{P'}) = f(\vec{P}) + (\vec{P'} - \vec{P})^T \vec{J}(\vec{P}) + \frac{1}{2}(\vec{P'} - \vec{P})^T \mathbf{H}(\vec{P})(\vec{P'} - \vec{P}) \tag{1}$$

The predicted change in network output according to the Taylor expansion:

$$\Delta \hat{f} = \hat{f}(\vec{P'}) - f(\vec{P}) \tag{2}$$

The true change in network output:

$$\Delta f = f(\vec{P'}) - f(\vec{P}) \tag{3}$$

Variance of the output change explained by the Taylor prediction using 2 and 3:

$$R^2_{Full} = corr(\Delta f, \Delta \hat{f})^2 \tag{4}$$

After the computation of R^2_{full}, terms were removed from the Taylor series. We note that both individual timepoints fed into the network and predictors are separated at this level. However, only the removal of predictors onto the quality of prediction was tested, not the importance of specific timepoints across predictors (i.e. all time points across a given predictor or interaction were removed to test the importance of the predictors, instead of removing all predictors at one time point to test the importance of a specific time point). For the removal of a given predictor both its linear (via J) and its squared contribution (via H) were subtracted from the full Taylor prediction while for the removal of interaction terms only the corresponding interaction term (via H) was subtracted. The following quantities were computed to arrive at the Taylor metric of single predictors P_n or

interactions of two predictors $P_{n,m}$ where P_n and J_n identify subvectors that contain all elements belonging to the specific predictor n while $H_{n,m}$ identifies all elements in the Hessian at the intersection of all rows belonging to the second derivatives with respect to predictor n and columns with respect to the predictor m:

Change predicted by only considering P_n:

$$\Delta\hat{f}_{P_n} = (\vec{P'_n} - \vec{P_n})^T J_n + \frac{1}{2}(\vec{P'_n} - \vec{P_n})^T \mathbf{H_{n,n}} (\vec{P'_n} - \vec{P_n}) \tag{5}$$

Change predicted when removing P_n using 2 and 5:

$$\Delta\hat{f}_{-P_n} = \Delta\hat{f} - \Delta\hat{f}_{P_n} \tag{6}$$

Change predicted when only considering interaction between P_n and P_m:

$$\Delta\hat{f}_{P_n,P_m} = \frac{1}{2}(\vec{P'_n} - \vec{P_n})^T \mathbf{H_{n,m}} (\vec{P'_m} - \vec{P_m}) + \frac{1}{2}(\vec{P'_m} - \vec{P_m})^T \mathbf{H_{m,n}} (\vec{P'_n} - \vec{P_n}) \tag{7}$$

Change predicted when removing P_nP_m interaction using 2 and 7:

$$\Delta\hat{f}_{-P_n,P_m} = \Delta\hat{f} - \Delta\hat{f}_{P_n,P_m} \tag{8}$$

Variance explained by Taylor prediction after removing P_n using 3 and 6:

$$R^2_{Pn} = corr(\Delta f, \Delta\hat{f}_{-P_n})^2 \tag{9}$$

Variance explained by Taylor prediction after removing P_nP_m interaction using 3 and 8:

$$R^2_{Pn,Pm} = corr(\Delta f, \Delta\hat{f}_{-P_n,P_m})^2 \tag{10}$$

The Taylor metric was then defined as:

$$T_{P_n} = 1 - R^2_{Full}/R^2_{Pn} \tag{11}$$

and

$$T_{P_n,P_m} = 1 - R^2_{Full}/R^2_{Pn,Pm} \tag{12}$$

respectively. For the zebrafish data we additionally calculated an overall "Behavior" Taylor metric in which all terms (regular and interaction) that belonged to any behavioral predictor were removed and the Taylor metric was subsequently calculated as outlined above.

Related python file in the repository: taylorDecomp.py

Characterizing the linearity and complexity of the transformation

Two metrics related to the nonlinearity of the function implemented by the CNN were calculated. A "curvature metric" which quantifies the magnitude of the influence of the second order derivative and the "nonlinearity coefficient" which approximates the error made by linearizing the network. It is important to note that given random inputs that are unrelated to the training data, it is very likely that the CNN will produce outputs that

nonlinearly depend on these inputs. Both metrics are therefore calculated with data that forms part of the training manifold[62]. We also note that the "curvature metric" does not in fact compute the curvature of the multi-dimensional function implemented by the CNN. Instead, the average magnitude of the vector induced by the *Hessian* on a unit magnitude step in the data-space is used as a proxy. With P, P' and f defined as above, the curvature metric is calculated according to:

$$C_f = \frac{1}{n} \sum_{\vec{P},\vec{P}'} \mathbf{H}(\vec{P})(\vec{P}' - \vec{P}) \tag{13}$$

The nonlinearity coefficient (NLC) was computed according to[33] quantifying:

$$NLC_{f,D} = \sqrt{\frac{\mathbb{E}_{\vec{P}\sim D}\left[Tr\left(J(\vec{P})^T Cov_{\vec{P}} J(\vec{P})\right)\right]}{Var_f}} = \sqrt{\frac{\mathbb{E}_{\vec{P},\vec{P}'\sim D}\left[Tr\left(J(\vec{P})^T(\vec{P}' - \vec{P})(\vec{P}' - \vec{P})^T J(\vec{P})\right)\right]}{Var_f}} \tag{14}$$

where D is the data distribution, Tr is the trace operator and $Cov_{\vec{P}}$ is the data covariance while Var_f is the output variance (since the output in the present case is a single value). The right form highlights that the NLC compares the variance of the linear approximation of the output $(J(\vec{P})^T(\vec{P}' - \vec{P}))$ with the true output variance.

Both metrics were combined in a simple logistic regression model to predict the probability that a given relationship between predictors (the CNN inputs) and neural activity (the CNN output) is nonlinear. This model was trained on the ground-truth data (Figure 3).

To compute a measure of complexity of the relationship between the predictors and neural activity, a Taylor expansion across the predictor average $\bar{P}$ was performed and its predictions were compared to the true model output again sampling predictors every second.

$$\hat{f}_{\bar{P}}(\vec{P}) = f(\bar{P}) + (\vec{P} - \bar{P})^T J(\bar{P}) + \frac{1}{2}(\vec{P} - \bar{P})^T \mathbf{H}(\vec{P} - \bar{P}) \tag{15}$$

Please note that the main difference in this formulation and the one above (for determining the set of contributing predictors) was that the *Jacobian* and *Hessian* are only calculated at one value, the predictor average. The formula above was then used to predict network outputs at P instead of changes in network outputs for a given change in P. The mean expansion score MES_f was then calculated as:

$$MES_f = corr(f, \hat{f}_{\bar{P}})^2 \tag{16}$$

The mean expansion score was thresholded at 0.5 (i.e., 50 % of the variance in the true network output was explained by the Taylor expansion around the data-mean). Then neurons were assigned to one of three respective complexity classes: **0** for neurons for which the nonlinearity model indicated linearity, **1** if the nonlinearity model indicated nonlinearity and $MES_f \geq 0.5$ or **2** if the neuron was deemed nonlinear and $MES_F < 0.5$.
Related python files in the repository: taylorDecomp.py, perf_nlc_nonlin.py and utilities.py

Comparison to spike-triggered analysis

To determine whether MINE could be used to extract system level characteristics akin to spike-triggered analysis a model was used to turn a randomly generated stimulus into a spike rate. The stimulus was either generated as a Gaussian white noise stimulus (by simply drawing values $S_i \sim \mathcal{N}(0,1)$ or as a smoothly varying random stimulus

as described in "Ground-truth datasets", page 26. The model consisted of two receptive fields (stimulus filters), RF_a and RF_v of length 50 timepoints, as well as two nonlinearities g_{sta} and g_{stc}. Given a stimulus s the rectified spike rate r was constructed according to:

$$r_{sta} = g_{sta}(s * RF_a) \tag{17}$$

$$r_{stc} = g_{stc}(s * RF_v) \tag{18}$$

$$r = max\left\{\frac{r_{sta}}{\sigma_{r_{sta}}} + \frac{r_{stc}}{\sigma_{r_{stc}}}, 0\right\} \tag{19}$$

, where $*$ denotes convolution and σ_x is the standard deviation of the indicated quantity. Normalization by the standard deviations was performed to ensure that not one filtered response dominated the overall spike rate.

The two nonlinearities were constructed such that g_{sta} was asymmetric and therefore led to a shift in the average spike rate, while g_{stc} was symmetric around 0 so that it influences the variance of the spike rate but not the average:

$$g_{sta}(x) = \frac{x}{1 + e^{-5x}} \tag{20}$$

$$g_{stc}(x) = 0.25 - \frac{e^x}{(1 + e^x)^2} \tag{21}$$

The model therefore generates a response function that upon spike triggered analysis (using appropriate white-noise stimuli with intensities drawn from a *Gaussian* distribution) will result in a spike-triggered average resembling RF_a while the eigenvector with the largest eigenvalue of the spike-triggered covariance matrix (ignoring RF_a itself) would resemble RF_v.

Examining equation 15 suggests that J has a similar influence on the network response as RF_a (the receptive field) has on neural responses. The projection of the input in direction of J will influence the average response, just like the projection of the stimulus along RF_a will influence the spike rate. Similarly, the eigenvectors of $\mathbf{H}$ are directions in stimulus space along which the variance of the response will be modified, similar to the eigenvectors of the spike triggered covariance matrix.

To test whether RF_a and RF_v could be recovered by MINE, the underlying CNN was subsequently fit using two thirds of the stimulus s as input and the generated spike rate r as target activity. The *Jacobian* J and the *Hessian* $\mathbf{H}$ were subsequently extracted from the network at the data mean. The *Jacobian* was subsequently compared to RF_a by measuring the cosine similarity (as in equation 32). Eigenvectors and eigenvalues were extracted from $\mathbf{H}$. The two eigenvectors with the largest eigenvalues were subsequently compared to J using the cosine similarity and the eigenvector with the lower similarity was subsequently compared to RF_v, again using the cosine similarity. The rationale for this was that RF_a affects both the averege spike rate and its variance and it will therefore generally be one of two eigenvectors with the largest eigenvalue.

To compare the results obtained from MINE to those obtained when using the data for spike-triggered analysis both a spike triggered average and a spike triggered covariance matrix were extracted. To this end vectors $\vec{x}_i$ were constructed so that for each time point i contained the stimulus history of 50 timepoints leading up to the time point i. Using the vector of spike rates r the spike triggered average and spike triggered covariance matrix were subsequently computed as weighted averages and weighted covariances according to:

$$\bar{x} = \frac{1}{n}\sum_i x_i \tag{22}$$

$$\overrightarrow{STA} = \frac{\sum_i (x_i - \bar{x}) r_i}{\sum_i r_i} \tag{23}$$

$$\mathbf{X_R} = \begin{vmatrix} (\vec{x}_0 - \bar{x}) r_0 \\ (\vec{x}_1 - \bar{x}) r_1 \\ \vdots \\ (\vec{x}_n - \bar{x}) r_n \end{vmatrix} \tag{24}$$

$$\mathbf{STC} = \frac{1}{\sum_i r_i} \mathbf{X_R}^T \mathbf{X_R} \tag{25}$$

The eigenvectors of **STC** with the two largest eigenvalues were directly compared to RF_v and the vector with the larger cosine similarity was chosen to be the estimate of RF_v.

For whitening, ZCA whitening[63] was performed to preserve the original rotation of the data. All $\vec{x}_i$ were adjusted by whitening the matrix $\mathbf{X}$ and then using its rows as the new $\vec{x}_i$ in all calculations above. The following steps were performed. The centered matrix $\mathbf{X}$ was created.

$$\mathbf{X} = \begin{vmatrix} \vec{x}_0 - \bar{x} \\ \vec{x}_1 - \bar{x} \\ \vdots \\ \vec{x}_n - \bar{x} \end{vmatrix} \tag{26}$$

The data covariance $\mathbf{C}$ was computed according to:

$$\mathbf{C} = \frac{1}{n}\mathbf{X}^T\mathbf{X} \tag{27}$$

Eigendecomposition was performed on $\mathbf{C}$:

$$\mathbf{C} = \mathbf{U}\mathbf{\Lambda}\mathbf{U}^T \tag{28}$$

where $\mathbf{U}$ is the matrix of eigenvectors and $\mathbf{\Lambda}$ is the diagonal matrix of eigenvalues. The whitening matrix $\mathbf{W}$ was subsequently computed according to:

$$\mathbf{W} = \mathbf{U}\mathbf{\Lambda}^{-1/2}\mathbf{U}^T \tag{29}$$

The whitening transform $\mathbf{W}$ was applied to $\mathbf{X}$:

$$\mathbf{X}_{ZCA} = (\mathbf{W}\mathbf{X}^T)^T \tag{30}$$

The whole procedure was repeated for 100 randomly generated stimuli to obtain different estimates for both MINE and spike-triggered analysis.

Related python file in the repository:cnn_sta_test.py

Clustering of neurons according to receptive field

Jacobians were extracted for each neuron at the data average and used as the receptive field (see section on comparison to spike-triggered analysis). Accordingly, the receptive field (RF_f) of a network was defined as:

$$RF_f = J(\bar{P}) \tag{31}$$

This is of course merely an approximation for nonlinear neurons. The first order derivative of a nonlinear function is not constant and therefore the receptive field will depend on the inputs.

To cluster zebrafish sensory neurons according to the receptive fields, the cosine of the angle between the receptive fields of different neurons (RF_i, RF_j, $i \neq j$) (and during the process, clusters) was computed (cosine similarity, CS):

$$CS_{RF_1,RF_2} = \frac{RF_1^T RF_2}{\|RF_1\|_2 \|RF_2\|_2} \tag{32}$$

Clustering was subsequently performed by a greedy approach as outlined in[64]. Briefly: All pairwise CS values were computed. Then, the pair of receptive fields with highest CS was aggregated into a cluster. The average RF in that cluster was computed. Subsequently all pairwise CS values, now containing that new cluster average, were computed again and the procedure was iteratively repeated until all receptive fields were clustered or no pairwise similarities $CS_{RF_1,RF_2} \geq 0.8$ were observed. The advantages of this clustering method are that it is a) deterministic and b) the number of clusters does not have to be specified beforehand.

In the case of the receptive fields on the mouse dataset, the matrix of pairwise cosine similarities was used to perform spectral clustering. Spectral clustering finds groups of strong connection (high CS in this case) within a similarity graph that are separated from other groups by weak connections (low CS). The rationale for changing clustering methods was computational speed which is much higher for spectral clustering.

Related python file in the repository: utilities.py

Ground-truth datasets

To test the capability of MINE to predict relationships between predictors and neural activity, four predictors were computer-generated. Two of these were generated by combining sine, square and triangle waves of random frequencies and amplitudes - simulating smoothly varying sensory inputs. The third predictor was generated as a Poisson process - simulating stochastic, unitary events such as bouts of movement. The last predictor was generated as a Poisson process as well but events were scaled with a random number drawn from a unit normal distribution with mean zero - simulating directed, stochastic movements.

Model responses were subsequently created by linearly or nonlinearly transforming individual predictors and combining them multiplicatively. After this step all responses were convolved with a "calcium kernel" and *i.i.d.* Gaussian noise was added to more closely resemble real world acquisition.

MINE was applied to these data: fitting a CNN on two-thirds of the data and calculating the correlation between the CNN prediction and the last third (test set) of the data. As a comparison, two linear regression models were fit to the data as well, again split into training and test sets. One of these models used raw predictors as inputs to predict the response (equivalent to the inputs given to MINE). The other used transformed predictors, convolved with the same calcium kernel applied to the responses above, along with all first-order interaction terms of the

predictors to predict the response. Correlations on the test data were subsequently used for comparisons.
Related python file in the repository: cnn_fit_test.py

To test the capability of MINE in predicting the nonlinearity of predictor-response relationships, for each test-case a new "sensory-like predictor" (introduced above) was generated. For the effect of varying nonlinearity (Figure 3 B-D) the predictor was transformed by a) squaring the hyperbolic tangent of the predictor, b) differencing the predictor or c) rectifying the predictor using a "softplus" transformation. These transformed versions were subsequently mixed to varying fractions with the original predictor to generate a response. The curvature and nonlinearity coefficient were then calculated on each mixture as indicated above. Linear correlations were also performed for some of the mixtures to illustrate the effect of both linear and nonlinear transformations on this metric. For ROC analysis of the nonlinearity metrics, a set of 500 by 5 random stimuli were generated. For each set four responses were generated depending only on the first of the five random stimuli (the others were used as noisy detractors to challenge the CNN fit). Two of the four responses were linear transformations: one was a differencing of the first predictor and the other was a convolution with a randomly generated double-exponential filter. The other two responses were randomized nonlinear transformations. The first was a variable-strength rectification by shifting and scaling a softplus function. The second was a random integer power between one and five of a hyperbolic tangent transformation of the predictor. The known category (linear vs. nonlinear) was subsequently used to calculate false- and true-positive rates at various thresholds for curvature, nonlinearity coefficient and our logistic nonlinearity model (see above) to perform ROC analysis.
Related python file in the repository: cnn_nonlin_test.py

Processing of data from Musall et al., 2019

The data for[40] was downloaded from the published dataset. Temporal components[7] were loaded from "interpVc.mat," spatial components from "Vc.mat" and predictors from "orgregData.mat" for each analyzed session. Predictors were shifted by five seconds relative to the activity traces. The 10 s long convolutional filters allowed for observing the influences of events five seconds into the past as well as preparatory activity for actions occuring in the next five seconds. MINE was subsequently used to determine the relationship between predictors and temporal components. The neural activity and the individual terms of the Taylor decomposition were mapped into pixel space. Then spatial maps were generated by computing Taylor metrics as above for each individual pixel timeseries. This was necessary since the components in "interpVc.mat" are highly linearly dependent making it impossible to carry measures of explained variance directly from "component space" into "pixel space." Receptive fields on the other hand were directly mapped into pixel space using the spatial component matrix as these are linear.
Related python files in the repository: processMusall.py, mine.py, plotMusall.py

Zebrafish experiments

All calcium imaging was performed on a custom-built 2-photon microscope at 2.4 Hz with a resolution of 0.78 μm/pixel with an average power of 12 mW at sample at 950 nm. The brain was divided into six overlapping regions, two divisions along the dorso-ventral axis and three divisions along the anterior-posterior axis. The most anterior point in the most anterior region was set to the olfactory epithelium while the most posterior point in the most posterior region was the start of the spinal cord.

Larval zebrafish expressing nuclear localized GCaMP in all neurons and mCherry in all glutamatergic neurons

(mitfa -/-; elavl3-H2B:GCaMP6s +/-; vGlut2a-mCherry) between five and seven days post fertilization were used for all experiments. Larval zebrafish were mounted and stimulated as previously described in[41] including scan stabilization as previously described to avoid artifacts associated with heating induced expansion of the preparation. The tail of the larva was free and tracked at 250 Hz using an online tracking method previously described[41]. Whenever our online tail-tracking missed a point along the tail an image was saved from which the tail was subsequently re-tracked offline as in[65]. Since the experiments were performed under a new microscope, we re-calculated a model translating stimulus laser strength to fish temperature as described in[41,66] and subsequently referred to as the "temperature model".

For each imaging-plane, 495 s of a "random wave" stimulus were generated by superimposing sine-waves of randomized amplitudes with periods between 13 s and 200 s. The stimuli were set to generate temperatures outside the noxious range between 22 °C and 30 °C.

For reticulospinal backfills Texas-Red Dextran 10,000 MW (Invitrogen D1828) at 12.5 % w/v was pressure injected into the spinal-cord of larval zebrafish under Tricaine (Syndel MS222) anesthesia. Injections were performed in fish expressing nuclear localized GCaMP-6s in all neurons (mitfa -/-; Elavl3-H2B:GCaMP6s) that were embedded in low-melting point agarose with a glass capillary having a 15 μm tip opening. Fish were unembedded after the procedure, woken from anesthesia and rested overnight. Fish were screened the next day for labeling, embedded again and imaged under the 2-photon microscope.

Processing of zebrafish data

Raw imaging data was pre-processed using CaImAn for motion correction and to identify units and extract associated calcium traces[45]. Laser stimulus data was recorded at 20 Hz and converted to temperatures using the temperature model (see above). The 250 Hz tail data was used to extract boutstarts based on the absolute angular derivative of the tail crossing an empirically determined threshold. The vigor was calculated as the sum of the absolute angular derivative across the swimbout, which is a metric of the energy of tail movement. The directionality was calculated as the sum of the angles of the tip of the tail across the swimbout, which is a metric of how one-sided the tail movement is. Extracted calcium data, stimulus temperatures and the three behavior metrics were subsequently interpolated / downsampled to a 5 Hz timebase. To reduce linear dependence the behavior metrics were subsequently orthogonalized using a modified Gram-Schmidt process.

Components outside the brain as well as very dim components inside the brain were identified by CaImAn. To avoid analyzing components that likely do not correspond to neurons labeled by GCaMP, all components for which the average brightness across imaging time was less than 0.1 photons were excluded. The number of analyzed components was thereby reduced from $706,543$ to $432,882$.

The CNN was subsequently fit to the remaining components using two thirds of imaging time as training data. For each neuron, the stimulus presented during recording and the observed behavioral metrics were used as inputs. Both the inputs and the calcium activity traces were standardized to zero mean and unit standard deviation before fitting. For every neuron in which the correlation $r \geq \sqrt{0.5}$ all discussed MINE metrics were extracted.

To generate the cyclically permuted controls, all calcium data was rotated forward by a third of the experiment length with wrap around. The rationale for constructing the control data in this manner was that it maintains the overall structure of the data but since both our stimuli and the elicited behavior are stochastic it should remove any true relationship between predictors and responses. The CNN was fit to the control data in the same manner as to the real data but no further metrics were extracted.

To identify which predictors significantly contributed to neural activity, R_{full}, R_{P1} and $R_{P1,P2}$ were bootstrapped, computing the Taylor metric for each variate. The standard deviation of the bootstrap variate was subsequently used to estimate confidence intervals according to a Normal distribution approximation. While confidence intervals could have been computed directly from the bootstrap variates this would have meant storing all samples for all neurons in questions to retain flexibility in significance cutoff. The significance level was set to $p < 0.05$ after Bonferroni correction across all fit neurons (effective p-value of $1.26x10^{-6}$). To be considered a driver of neural activity the Taylor metric had to be larger than 0.1 at this significance threshold.

The linear comparison model was constructed akin to[7]. All predictors were timeshifted by between 0 and 50 timesteps for a total of 200 predictor inputs to the regression model. This setup emulates the convolutional filters present in the CNN used by MINE. A Modified-Gram-Schmidt process was used for orthogonalization to avoid the problems of QR decomposition observed in cases of near singular design matrices (singular up to the used floating-point precision). Ridge regression[9] was used to improve generalization. Setting the ridge penalty to 10^{-4} led to a 25-50 % increase of identified neurons on small test datasets compared to a standard regression model.

All zebrafish stacks (except those acquired after spinal backfills) were first registered to an internal HuC-H2B:GCaMP6s reference using CMTK[67]. A transformation from the internal reference to the Z-Brain[46] was subsequently performed using ANTS[68]. Micrometer coordinates transformed into Z-Brain pixel coordinates together with region masks present in the Z-Brain were subsequently used to assign all neurons to brain regions.

To identify Z-brain regions in which a specific functional type is enriched, the following process was used. The total number of all types under consideration in each region was computed (i.e., if the comparison was between stimulus, behavior and mixed selectivity neurons all these neurons were summed but if the comparison was between different classes of behavior-related neurons only those were summed). Subsequently, the fraction of the type of interest among the total in each region was computed (plotted as bars in Figures S5 and S6). Similarly, an overall fraction of the type across the whole brain was computed (plotted as vertical blue lines in Figures S5 and S6). This overall fraction was used to simulate a 95 % confidence interval of observed fractions given the total number of neurons considered in each region (horizontal blue lines in Figures S5 and S6). Any true fraction above this interval was considered significant enrichment. We note that exact confidence intervals could have been computed from the binomial distribution, however the simulation approach allowed us to use the same method when computing confidence intervals around the average complexity of receptive field clusters.

For reticulospinal backfill experiments, neurons labeled by the backfill were manually annotated. This annotation was used to classify neurons (CaImAn identified components) as reticulospinal.

Experiments were excluded from further analysis for the following reasons: five fish died during functional imaging; eight fish could not be registered to the reference; tail tracking failed during the acquisition of six fish and during the acquisition of four fish problems in the acquisitoin hardware led to imaging artifacts.

Related python files in the repository: rwave_data_fit.py, rwave_decompose.py, utilities.py

Author Contributions

Conceptualization: MH, JDC
Methodology: JDC, MH, KAB
Software: MH, JDC
Analysis: MH, JDC, KAB
Experiments: KAB, SS

Supervision, Resources and Funding Acquisition: MH
Writing: MH, JDC, KAB

Acknowledgements

We thank Andrew D. Bolton and Bradley Cutler for critical comments on the manuscript and the "Zebrafish Neural Circuits & Behavior" seminar community for early stage feedback and discussion. This work was supported by funding from The Ohio State University and NIH BRAIN Initiative R01NS123887 through NINDS and NIDA to MH.

References

[1] Engert, F. The big data problem: turning maps into knowledge. *Neuron* **83**, 1246–1248 (2014).

[2] Musall, S., Urai, A. E., Sussillo, D. & Churchland, A. K. Harnessing behavioral diversity to understand neural computations for cognition. *Curr. Opin. Neurobiol.* **58**, 229–238 (2019).

[3] Urai, A. E., Doiron, B., Leifer, A. M. & Churchland, A. K. Large-scale neural recordings call for new insights to link brain and behavior. *Nat. Neurosci.* (2022).

[4] Miri, A., Daie, K., Burdine, R. D., Aksay, E. & Tank, D. W. Regression-based identification of behavior-encoding neurons during large-scale optical imaging of neural activity at cellular resolution. *J. Neurophysiol.* **105**, 964–980 (2011).

[5] Harvey, C. D., Coen, P. & Tank, D. W. Choice-specific sequences in parietal cortex during a virtual-navigation decision task. *Nature* **484**, 62–68 (2012).

[6] Portugues, R., Feierstein, C. E., Engert, F. & Orger, M. B. Whole-brain activity maps reveal stereotyped, distributed networks for visuomotor behavior. *Neuron* **81**, 1328–1343 (2014).

[7] Musall, S., Kaufman, M. T., Juavinett, A. L., Gluf, S. & Churchland, A. K. Single-trial neural dynamics are dominated by richly varied movements. *Nat. Neurosci.* **22**, 1677–1686 (2019).

[8] Poggio, T. A theory of how the brain might work. *Cold Spring Harb. Symp. Quant. Biol.* **55**, 899–910 (1990).

[9] Friedman, J., Hastie, T., Tibshirani, R. & Others. *The elements of statistical learning*, vol. 1 (Springer series in statistics New York, 2001).

[10] Eliasmith, C. & Anderson, C. H. *Neural Engineering (Computational Neuroscience Series): Computational, Representation, and Dynamics in Neurobiological Systems* (MIT Press, Cambridge, MA, USA, 2002).

[11] Niven, J. E. & Laughlin, S. B. Energy limitation as a selective pressure on the evolution of sensory systems. *J. Exp. Biol.* **211**, 1792–1804 (2008).

[12] Zabihi, M. *et al.* Non-linearity matters: a deep learning solution to the generalization of hidden brain patterns across population cohorts (2021).

[13] James, G., Witten, D., Hastie, T. & Tibshirani, R. *An Introduction to Statistical Learning* (Springer US, 2013).

[14] Hornik, K., Stinchcombe, M. & White, H. Multilayer feedforward networks are universal approximators. *Neural Netw.* **2**, 359–366 (1989).

[15] Gorban, A. N. & Wunsch, D. C. The general approximation theorem. In *1998 IEEE International Joint Conference on Neural Networks Proceedings. IEEE World Congress on Computational Intelligence (Cat. No.98CH36227)*, vol. 2, 1271–1274 vol.2 (1998).

[16] Krogh & Hertz. A simple weight decay can improve generalization. *Adv. Neural Inf. Process. Syst.* (1991).

[17] Srivastava, N., Hinton, G., Krizhevsky, A., Sutskever, I. & Salakhutdinov, R. Dropout: A simple way to prevent neural networks from overfitting. *J. Mach. Learn. Res.* **15**, 1929–1958 (2014).

[18] Lehky, S. R., Sejnowski, T. J. & Desimone, R. Predicting responses of nonlinear neurons in monkey striate cortex to complex patterns. *J. Neurosci.* **12**, 3568–3581 (1992).

[19] Lau, B., Stanley, G. B. & Dan, Y. Computational subunits of visual cortical neurons revealed by artificial neural networks. *Proc. Natl. Acad. Sci. U. S. A.* **99**, 8974–8979 (2002).

[20] Prenger, R., Wu, M. C.-K., David, S. V. & Gallant, J. L. Nonlinear V1 responses to natural scenes revealed by neural network analysis. *Neural Netw.* **17**, 663–679 (2004).

[21] Ukita, J., Yoshida, T. & Ohki, K. Characterisation of nonlinear receptive fields of visual neurons by convolutional neural network. *Sci. Rep.* **9**, 3791 (2019).

[22] Keshishian, M. *et al.* Estimating and interpreting nonlinear receptive field of sensory neural responses with deep neural network models. *Elife* **9** (2020).

[23] Lucas, A. *et al.* Neural networks for modeling neural spiking in S1 cortex. *Front. Syst. Neurosci.* **13**, 13 (2019).

[24] Ramachandran, P., Zoph, B. & Le, Q. V. Searching for activation functions (2017). 1710.05941.

[25] Benjamin, A. S. *et al.* Modern machine learning as a benchmark for fitting neural responses. *Front. Comput. Neurosci.* **12**, 56 (2018).

[26] Hubel, D. H. & Wiesel, T. N. Receptive fields and functional architecture of monkey striate cortex. *J. Physiol.* **195**, 215–243 (1968).

[27] Churchland, P. S. & Sejnowski, T. J. *The Computational Brain* (MIT Press, 1994).

[28] Carandini, M. *et al.* Do we know what the early visual system does? *J. Neurosci.* **25**, 10577–10597 (2005).

[29] Thompson, J. A., Costabile, J. D. & Felsen, G. Mesencephalic representations of recent experience influence decision making. *Elife* **5** (2016).

[30] Pho, G. N., Goard, M. J., Woodson, J., Crawford, B. & Sur, M. Task-dependent representations of stimulus and choice in mouse parietal cortex. *Nat. Commun.* **9**, 2596 (2018).

[31] Marder, E. & Abbott, L. F. Theory in motion. *Curr. Opin. Neurobiol.* **5**, 832–840 (1995).

[32] Shamir, M. & Sompolinsky, H. Implications of neuronal diversity on population coding. *Neural Comput.* **18**, 1951–1986 (2006).

[33] Philipp, G. & Carbonell, J. G. The nonlinearity coefficient - predicting generalization in deep neural networks (2018). 1806.00179.

[34] Philipp, G. The nonlinearity coefficient - a practical guide to neural architecture design (2021). 2105.12210.

[35] Dayan, P., Abbott, L. F. & Others. *Theoretical neuroscience*, vol. 806 (Cambridge, MA: MIT Press, 2001).

[36] Marmarelis, P. Z. & Marmarelis, V. Z. *Analysis of Physiological Systems* (Springer Verlag, 1978).

[37] Rieke, F., Warland, D., Van Steveninck, R. D. R. & Bialek, W. *Spikes: Exploring the Neural Code* (MIT Press, 1999).

[38] Schwartz, O., Pillow, J. W., Rust, N. C. & Simoncelli, E. P. Spike-triggered neural characterization. *J. Vis.* **6**, 484–507 (2006).

[39] Gollisch, T. & Meister, M. Modeling convergent ON and OFF pathways in the early visual system. *Biol. Cybern.* **99**, 263–278 (2008).

[40] Churchland, A. K., Musall, S., Kaufman, M. T., Juavinett, A. L. & Gluf, S. Dataset of "single-trial neural dynamics are dominated by richly varied movements" (2019). Title of the publication associated with this dataset: Single-trial neural dynamics are dominated by richly varied movements.

[41] Haesemeyer, M., Robson, D. N., Li, J. M., Schier, A. F. & Engert, F. A brain-wide circuit model of Heat-Evoked swimming behavior in larval zebrafish. *Neuron* **98**, 817–831.e6 (2018).

[42] Haesemeyer, M., Schier, A. F. & Engert, F. Convergent temperature representations in artificial and biological neural networks. *Neuron* **103**, 1123–1134.e6 (2019).

[43] Freeman, J. *et al.* Mapping brain activity at scale with cluster computing. *Nat. Methods* **11**, 941–950 (2014).

[44] Satou, C. *et al.* Transgenic tools to characterize neuronal properties of discrete populations of zebrafish neurons. *Development* **140**, 3927–3931 (2013).

[45] Giovannucci, A. *et al.* CaImAn an open source tool for scalable calcium imaging data analysis. *Elife* **8** (2019).

[46] Randlett, O. *et al.* Whole-brain activity mapping onto a zebrafish brain atlas. *Nat. Methods* **12**, 1039–1046 (2015).

[47] Schmid, A., Pyrski, M., Biel, M., Leinders-Zufall, T. & Zufall, F. Grueneberg ganglion neurons are finely tuned cold sensors. *J. Neurosci.* **30**, 7563–7568 (2010).

[48] Kludt, E., Okom, C., Brinkmann, A. & Schild, D. Integrating temperature with odor processing in the olfactory bulb. *J. Neurosci.* **35**, 7892–7902 (2015).

[49] Fleischer, J. The grueneberg ganglion: signal transduction and coding in an olfactory and thermosensory organ involved in the detection of alarm pheromones and predator-secreted kairomones. *Cell Tissue Res.* **383**, 535–548 (2021).

[50] Saad, F. A., Cusumano-Towner, M. F., Schaechtle, U., Rinard, M. C. & Mansinghka, V. K. Bayesian synthesis of probabilistic programs for automatic data modeling. *Proc. ACM Program. Lang.* **3**, 1–32 (2019).

[51] Frey, M. *et al.* Interpreting wide-band neural activity using convolutional neural networks. *Elife* **10** (2021).

[52] Schneider, S., Lee, J. H. & Mathis, M. W. Learnable latent embeddings for joint behavioral and neural analysis (2022). 2204.00673.

[53] Rumelhart, D. E., McClelland, J. L. & Pdp research group. *Parallel distributed processing*, vol. 1 (MIT press Cambridge, MA, 1987).

[54] Samek, W., Montavon, G., Vedaldi, A., Hansen, L. K. & Müller, K.-R. (eds.) *Explainable AI: Interpreting, Explaining and Visualizing Deep Learning*. Lecture Notes in Computer Science (Springer International Publishing, 2019).

[55] Binder, A., Bach, S., Montavon, G., Müller, K.-R. & Samek, W. Layer-Wise relevance propagation for deep neural network architectures. In *Information Science and Applications (ICISA) 2016*, 913–922 (Springer Singapore, 2016).

[56] Novak, R., Bahri, Y., Abolafia, D. A., Pennington, J. & Sohl-Dickstein, J. Sensitivity and generalization in neural networks: an empirical study (2018). 1802.08760.

[57] Sandler, R. A. & Marmarelis, V. Z. Understanding spike-triggered covariance using wiener theory for receptive field identification. *J. Vis.* **15**, 16 (2015).

[58] Paninski, L. Convergence properties of some spike-triggered analysis techniques. *Adv. Neural Inf. Process. Syst.* (2002).

[59] Aljadeff, J., Segev, R., Berry, M. J., 2nd & Sharpee, T. O. Spike triggered covariance in strongly correlated gaussian stimuli. *PLoS Comput. Biol.* **9**, e1003206 (2013).

[60] Kawashima, T., Zwart, M. F., Yang, C.-T., Mensh, B. D. & Ahrens, M. B. The serotonergic system tracks the outcomes of actions to mediate Short-Term motor learning. *Cell* **167**, 933–946.e20 (2016).

[61] Abadi, M. *et al.* Tensorflow: a system for large-scale machine learning. In *OSDI*, vol. 16, 265–283 (2016).

[62] Raghu, M., Poole, B., Kleinberg, J., Ganguli, S. & Sohl-Dickstein, J. On the expressive power of deep neural networks. In Precup, D. & Teh, Y. W. (eds.) *Proceedings of the 34th International Conference on Machine Learning*, vol. 70 of *Proceedings of Machine Learning Research*, 2847–2854 (PMLR, 2017).

[63] Bell, A. J. & Sejnowski, T. J. The "independent components" of natural scenes are edge filters. *Vision Res.* **37**, 3327–3338 (1997).

[64] Bianco, I. H. & Engert, F. Visuomotor transformations underlying hunting behavior in zebrafish. *Curr. Biol.* **25**, 831–846 (2015).

[65] Mearns, D. S., Donovan, J. C., Fernandes, A. M., Semmelhack, J. L. & Baier, H. Deconstructing hunting behavior reveals a tightly coupled Stimulus-Response loop. *Curr. Biol.* **30**, 54–69.e9 (2020).

[66] Haesemeyer, M., Robson, D. N., Li, J. M., Schier, A. F. & Engert, F. The structure and timescales of heat perception in larval zebrafish. *Cell Syst* **1**, 338–348 (2015).

[67] Rohlfing, T. & Maurer, C. R., Jr. Nonrigid image registration in shared-memory multiprocessor environments with application to brains, breasts, and bees. *IEEE Trans. Inf. Technol. Biomed.* **7**, 16–25 (2003).

[68] Avants, Tustison & Song. Advanced normalization tools (ANTS). *Insight J.* (2009).

Supplemental Figures

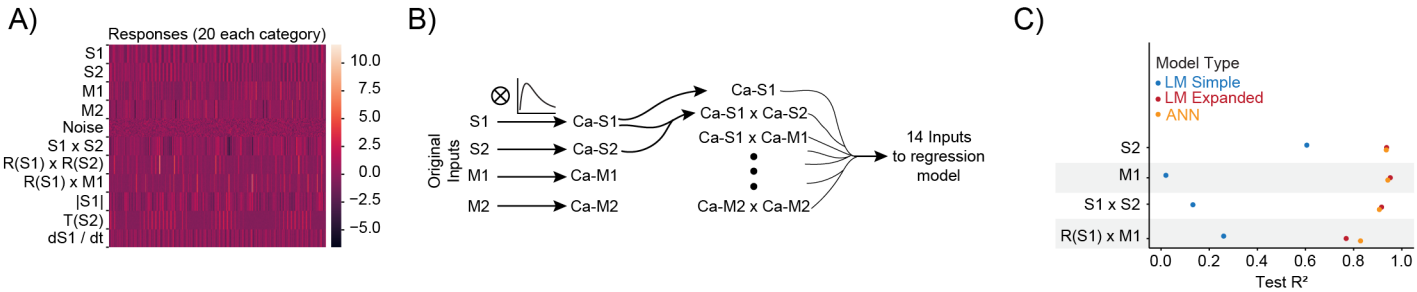

Figure S1: (Related to Figure 1) MINE accommodates a large set of predictor-activity relationships
A) Heatmap of all generated responses in the ground truth dataset 220 total across 11 groups including pure noise responses.
B) Schematic of the expanded linear model which includes all first-order interaction terms and convolution with the true "calcium kernel" used in the ground-truth dataset. We note however that since multiplication is a nonlinear operation, the generation of the interaction terms after convolution is not exactly the same as the generation of the interaction terms in the ground-truth dataset (where convolution occurs after the multiplication).
C) For data not shown in Figure 1F), fraction of explained variance on test data for a simple linear regression model (blue), a linear regression model including first-order interaction terms and calcium kernel convolution (red) as well as the CNN model (orange). Each dot represents the average across 20 models. While the standard deviation is represented by a dash, it is smaller than the dot size in all cases and therefore not visible in the graph.

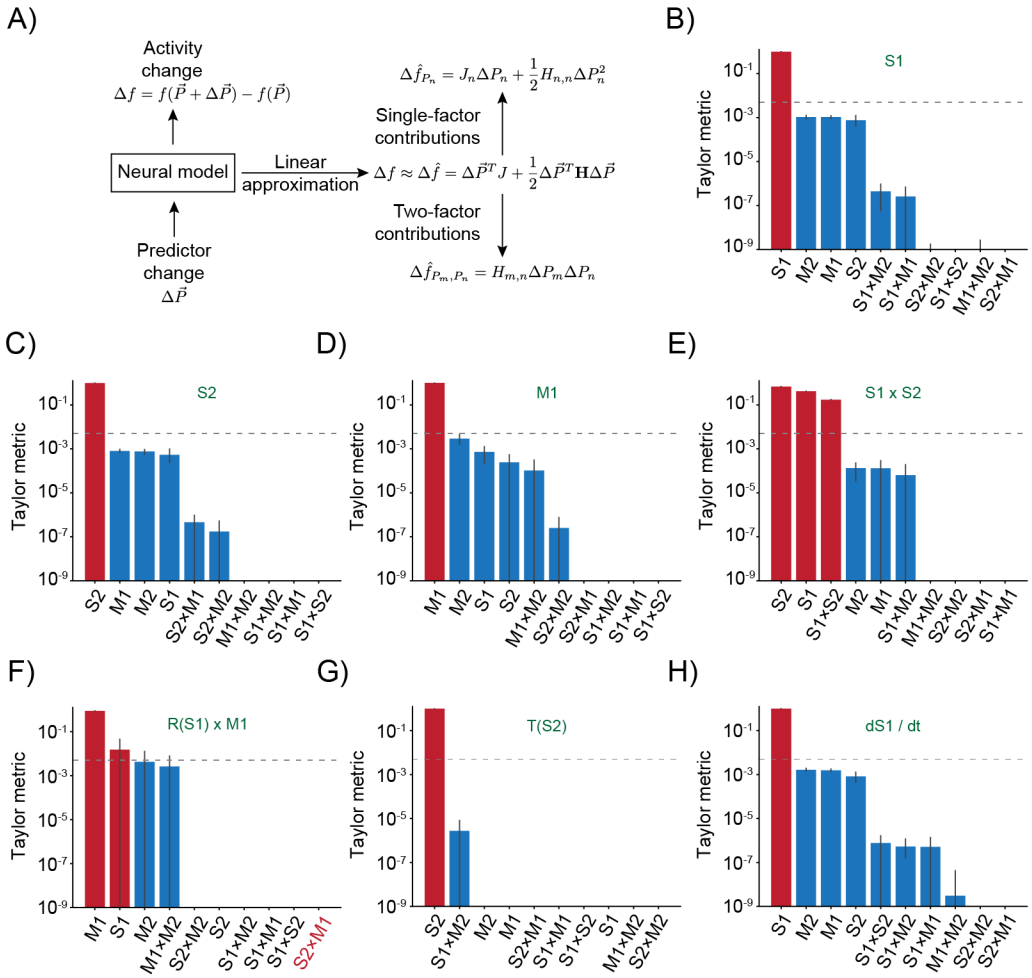

Figure S2: (Related to Figure 2) Taylor decomposition reveals contributing single factors and interactions
A) Schematic highlighting key parts of the decomposition that isolates individual factors.
B-H) Remaining responses and associated Taylor metrics. Red bars indicate predictors that are expected to contribute, blue bars those that should not contribute. Dashed line indicates a threshold at 5x10-3 which identifies true contributing predictors in all cases. Error bars are 95 % bootstrap confidence intervals.
B) Response type only depending on S1 predictor.
C) Response type only depending on S2 predictor.
D) Response type only depending on M1 predictor.
E) Response to multiplicative interaction of S1 and S2 predictors.
F) Response to multiplicative interaction of rectified S1 and original M1 predictor. Note that the S2xM1 term which should contribute has the lowest metric.
G) Response thresholding S2 predictor.
H) Response to the derivative of S1.

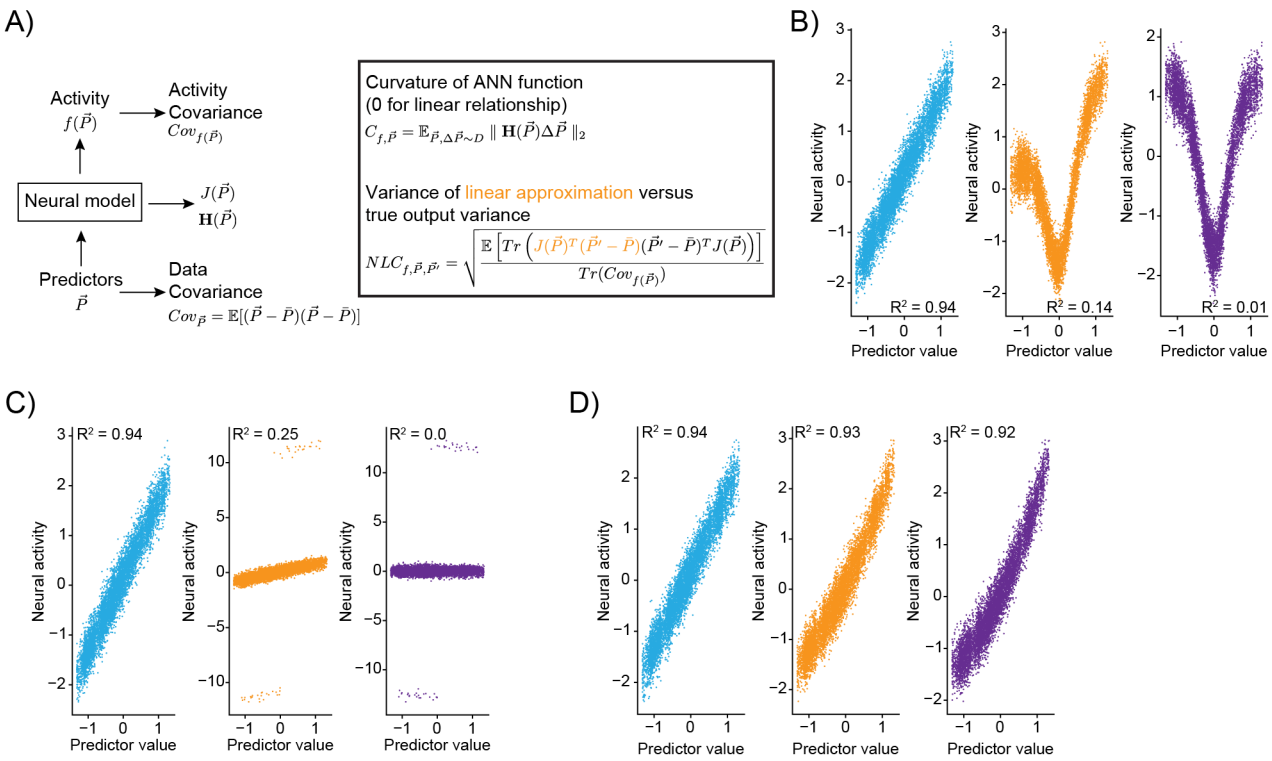

Figure S3: (related to Figure 3) Measures of nonlinearity separate linear from non-linear predictor-activity relationships

A) Calculation of the nonlinearity metrics curvature and NLC and their relationship to the data and neural model.

B-D) Linear correlations between input predictors and ground-truth data "neural activity" for different mixtures. Colors of correlation plots correspond to colored examples in Figure 3.

B) Mixing of a nonlinear $y = \tanh(2x)^2$ and linear transformation of the predictor.

C) Mixing of a predictor and its derivative.

D) Mixing of a predictor and a rectified $y = ln(e^x + 1)$ version of the predictor.

Note: Outputs are recentered to 0 mean and 1 standard deviation before being passed to the ANN and before being plotted here.

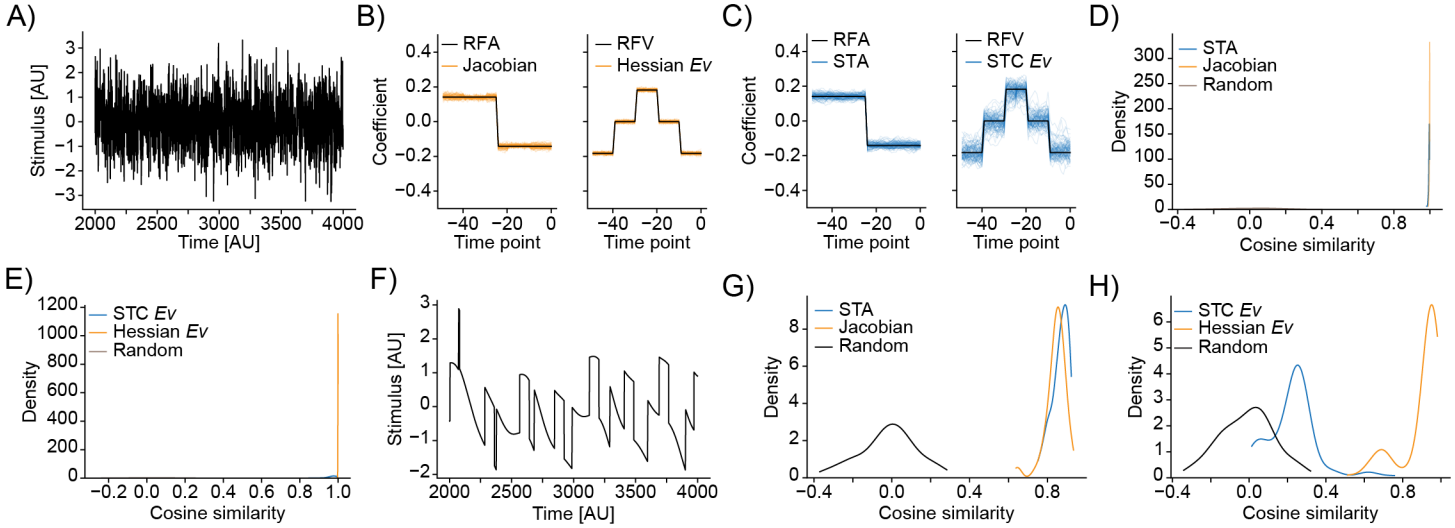

Figure S4: Comparison of MINE and spike-triggered analysis

A Example section of a white-noise stimulus trace used in the analysis.

B) Approximations of the receptive field affecting the average spike rate (RFA) using the *Jacobian* of the CNN fit by MINE (left) and of the receptive field affecting the variance of the spike rate (RFV) using the eigenvector of the *Hessian* of the CNN fit by Mine. Black lines are the true receptive fields, thin orange lines are derived approximations across $N = 100$ randomly generated, white-noise stimuli.

C) Same as B) but approximations derived using spike-triggered analysis. The spike-triggered average (STA) left and eigenvector of the spike-triggered covariance matrix (STC) right are compared to the RFA and RFV respectively.

D) Across $N = 100$ randomly generated, white-noise stimuli, distribution of cosine similarities between the RFA and the MINE *Jacobian* (orange), the spike-triggered average (blue) and a random vector (brown).

E) As in D) but distribution of cosine similarities between the RFC and the MINE *Hessian* eigenvector (orange), the eigenvector of the spike-triggered covariance matrix (blue) and a random vector (brown).

F) Example section of one smoothly varying stimulus trace used in the analysis.

G) Across $N = 100$ randomly generated, smoothly varying stimuli, distribution of cosine similarities between the RFA and the MINE *Jacobian* (orange), the spike-triggered average (blue) after the stimulus has been pre-whitened using ZCA whitening, and a random vector (black).

H) As in G) but distribution of cosine similarities between the RFC and the MINE *Hessian* eigenvector (orange), the eigenvector of the spike-triggered covariance matrix (blue) after the stimulus has been pre-whitened using ZCA whitening, and a random vector (black).

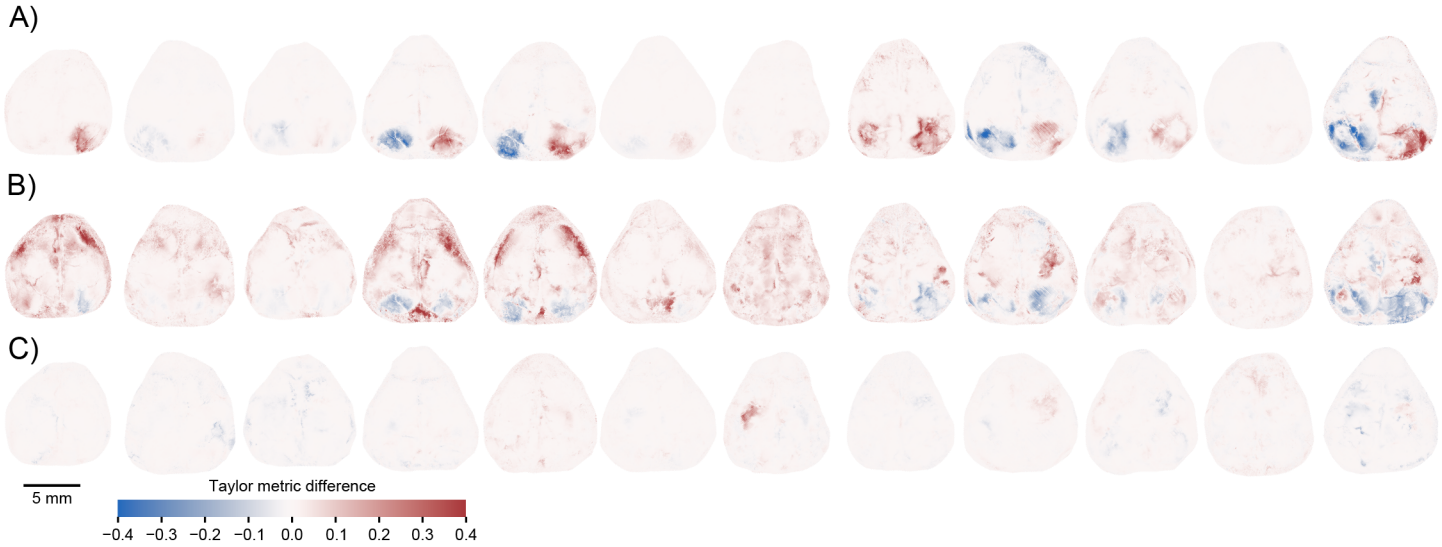

Figure S5: (related to Figure 5) MINE identifies cortical features of sensori-motor processing during a learned task
A) Per pixel Taylor-metric scores for the right visual stimulus ("rVisStim") subtracted from those of the left visual stimulus ("lVisStim") for the twelve mice not shown in Figure 4F.
B) As A) but the sum of the visual stimulus Taylor-metrics subtracted from the Taylor metrics of the whisking predictor for the same twelve mice.
C) As A) but the Taylor-metric for the right grab predictor has been subtracted from the Taylor-metric of the left grab predictor.

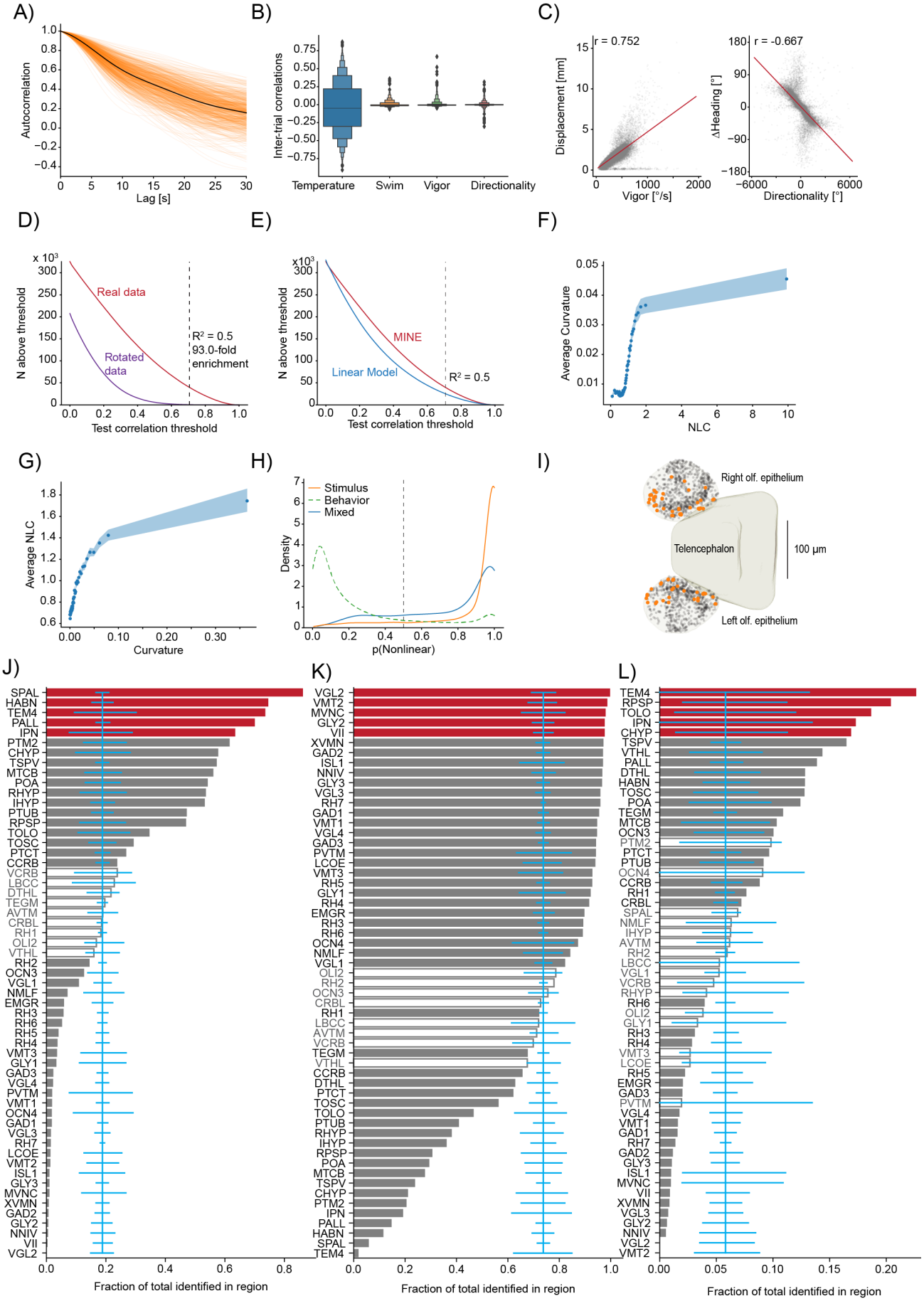
A)
Autocorrelation
Lag [s]
B)
Inter-trial correlations
Temperature
Swim
Vigor
Directionality
C)
r = 0.752
Displacement [mm]
Vigor [°/s]
r = -0.667
ΔHeading [°]
Directionality [°]
D)
x 10³
N above threshold
Real data
Rotated data
R² = 0.5
93.0-fold enrichment
Test correlation threshold
E)
x10³
N above threshold
MINE
Linear Model
R² = 0.5
Test correlation threshold
F)
Average Curvature
NLC
G)
Average NLC
Curvature
H)
Stimulus
Behavior
Mixed
Density
p(Nonlinear)
I)
Right olf. epithelium
Telencephalon
100 µm
Left olf. epithelium
J)
SPAL
HABN
TEM4
PALL
IPN
PTM2
CHYP
TSPV
MTCB
POA
RHYP
IHYP
PTUB
RPSP
TOLO
TOSC
PTCT
CCRB
VCRB
LBCC
DTHL
TEGM
AVTM
CRBL
RH1
OLI2
VTHL
RH2
OCN3
VGL1
NMLF
EMGR
RH3
RH6
RH5
RH4
VMT3
GLY1
GAD3
VGL4
PVTM
VMT1
OCN4
GAD1
VGL3
RH7
LCOE
VMT2
ISL1
GLY3
MVNC
XVMN
GAD2
GLY2
NNIV
VII
VGL2
Fraction of total identified in region
K)
VGL2
VMT2
MVNC
GLY2
VII
XVMN
GAD2
ISL1
NNIV
GLY3
VGL3
RH7
GAD1
VMT1
VGL4
GAD3
PVTM
LCOE
VMT3
RH5
GLY1
RH4
EMGR
RH3
RH6
OCN4
NMLF
VGL1
OLI2
RH2
OCN3
CRBL
RH1
LBCC
AVTM
VCRB
TEGM
VTHL
CCRB
DTHL
PTCT
TOSC
TOLO
PTUB
RHYP
IHYP
RPSP
POA
MTCB
TSPV
CHYP
PTM2
IPN
PALL
HABN
SPAL
TEM4
Fraction of total identified in region
L)
TEM4
RPSP
TOLO
IPN
CHYP
TSPV
VTHL
PALL
DTHL
HABN
TOSC
POA
TEGM
MTCB
OCN3
PTM2
PTCT
PTUB
OCN4
CCRB
RH1
CRBL
SPAL
NMLF
IHYP
AVTM
RH2
LBCC
VGL1
VCRB
RHYP
RH6
OLI2
GLY1
RH3
RH4
VMT3
LCOE
RH5
EMGR
GAD3
PVTM
VGL4
VMT1
GAD1
RH7
GAD2
GLY3
ISL1
MVNC
VII
XVMN
VGL3
GLY2
NNIV
VGL2
VMT2
Fraction of total identified in region

Figure S6: (related to Figure 6) Using MINE to probe larval zebrafish thermoregulatory circuits

A) Across all imaged planes of all fish (orange lines) the autocorrelation of the provided stimulus. Black line is the average across all planes ($N = 750$).

B) Boxenplot of the inter-trial correlations for the temperature stimulus and the three behavioral metrics. Note, that we use the term "trial" here to denote equal thirds of the imaging period in one plane which allows us to use two "trials" for training and one for testing the fit ANNs and linear models. Since there is no repetitive structure the term "trial" is used loosely here.

C) Scatterplot and correlations on free-swimming data between the "Vigor" (left) and "Directionality" (right) tail metrics and free-swimming actual displacements (left) and actual heading-angle changes (right).

D) For different test-correlation thresholds (x-axis) the number of units for which the ANN has above-threshold test-correlations (considered identified by the ANN). Red curve, real data, purple curve rotated data where the calcium activity has been rotated by one "trial" with respect to the stimulus and behavior predictors. Black dashed line indicates the threshold used in the paper and lists the ratio of identified units in the real versus rotated data.

E) Same as D) but comparing the ANN (replot from D) with above threshold correlations for the linear model. Dashed line again indicates the threshold used in the paper.

F) Across all identified units, the average curvature metric according to the NLC metric. Data divided into 50 even-sized quantiles based on the NLC metric. Band is bootstrap standard error across all data-points within the quantile.

G) Across all identified units, the average NLC metric according to the curvature metric. Data divided into 50 even-sized quantiles based on the curvature metric. Band is bootstrap standard error across all data-points within the quantile.

H) For stimulus (orange), behavior (dashed green) and mixed selectivity (blue) neurons the distribution of nonlinearity probabilities.

I) Identified temperature encoding units in the olfactory epithelium. Orange dots are units that according to Taylor metric encode the stimulus while black dots represent units within the epithelium that were not identified by the ANN to highlight medial enrichment of temperature encoding neurons.

J-L) For each analyzed Z-Brain region the units encoding the stimulus **(J)**, behavior **(K)** or both **(L)** as a fraction of the sum across these three groups. Red bars highlight the top five regions with significant enrichment plotted in Figure 5H. Open bars and gray text highlight those regions that are neither significantly enriched nor significantly depleted of the given type. Blue line is the expected (average) fraction of the given type. Error-bars mark a boot-strapped 95% confidence interval around the average of possible observed fractions if the region would in fact contain the expected fraction of units. Bars for which the top is above this confidence interval are therefore significantly enriched, bars for which the top is below are significantly depleted. Note that regions in which less than 50 neurons total across the listed categories have been identified, have not been analyzed for enrichment. Region abbreviations are listed in Table S1.

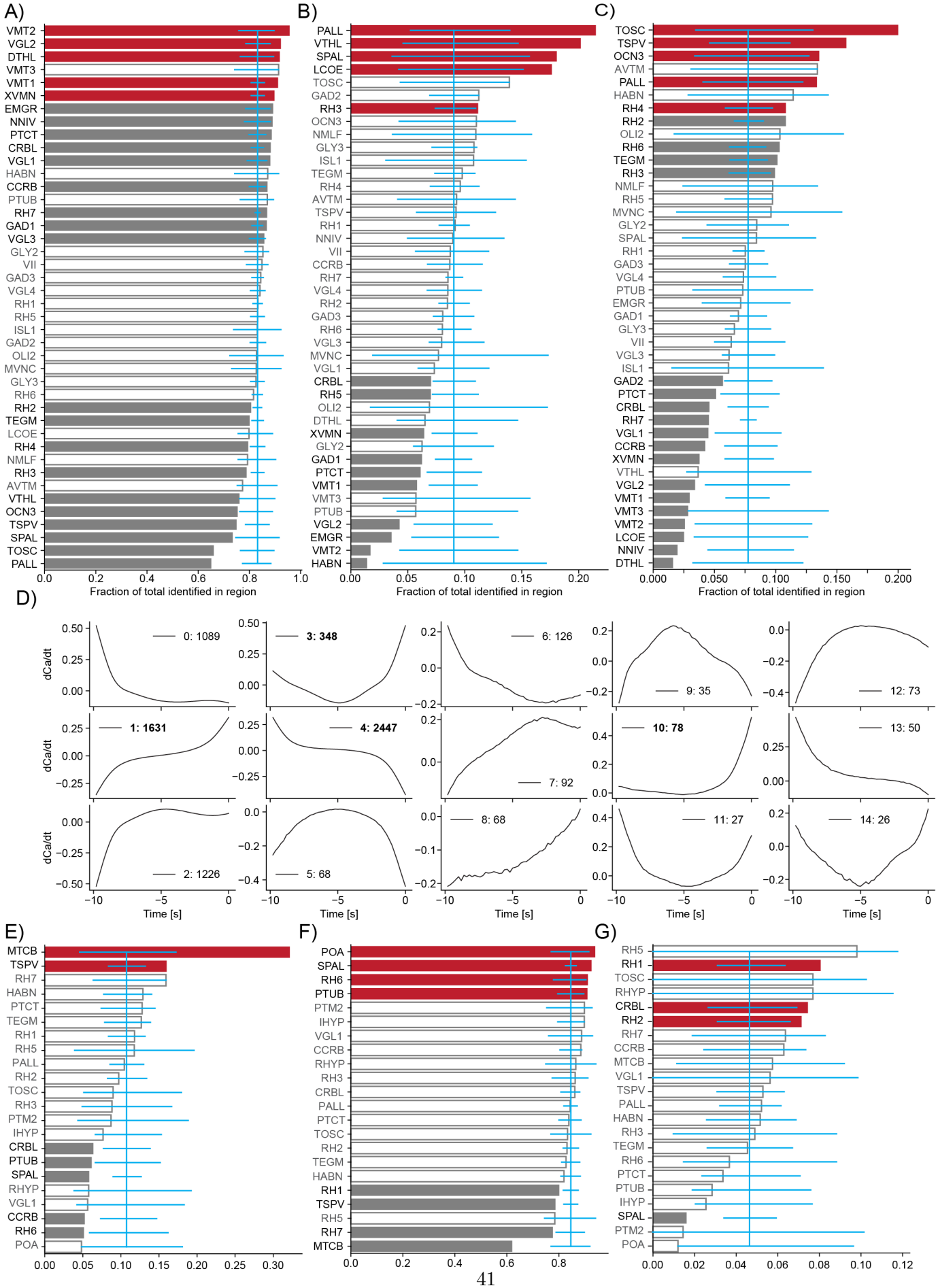
A)
VMT2
VGL2
DTHL
VMT3
VMT1
XVMN
EMGR
NNIV
PTCT
CRBL
VGL1
HABN
CCRB
PTUB
RH7
GAD1
VGL3
GLY2
VII
GAD3
VGL4
RH1
RH5
ISL1
GAD2
OLI2
MVNC
GLY3
RH6
RH2
TEGM
LCOE
RH4
NMLF
RH3
AVTM
VTHL
OCN3
TSPV
SPAL
TOSC
PALL
0.0 0.2 0.4 0.6 0.8 1.0
Fraction of total identified in region
B)
PALL
VTHL
SPAL
LCOE
TOSC
GAD2
RH3
OCN3
NMLF
GLY3
ISL1
TEGM
RH4
AVTM
TSPV
RH1
NNIV
VII
CCRB
RH7
VGL4
RH2
GAD3
RH6
VGL3
MVNC
VGL1
CRBL
RH5
OLI2
DTHL
XVMN
GLY2
GAD1
PTCT
VMT1
VMT3
PTUB
VGL2
EMGR
VMT2
HABN
0.00 0.05 0.10 0.15 0.20
Fraction of total identified in region
C)
TOSC
TSPV
OCN3
AVTM
PALL
HABN
RH4
RH2
OLI2
RH6
TEGM
RH3
NMLF
RH5
MVNC
GLY2
SPAL
RH1
GAD3
VGL4
PTUB
EMGR
GAD1
GLY3
VII
VGL3
ISL1
GAD2
PTCT
CRBL
RH7
VGL1
CCRB
XVMN
VTHL
VGL2
VMT1
VMT3
VMT2
LCOE
NNIV
DTHL
0.000 0.050 0.100 0.150 0.200
Fraction of total identified in region
D)
dCa/dt
0: 1089
3: 348
6: 126
9: 35
12: 73
1: 1631
4: 2447
7: 92
10: 78
13: 50
2: 1226
5: 68
8: 68
11: 27
14: 26
−10 −5 0
Time [s]
E)
MTCB
TSPV
RH7
HABN
PTCT
TEGM
RH1
RH5
PALL
RH2
TOSC
RH3
PTM2
IHYP
CRBL
PTUB
SPAL
RHYP
VGL1
CCRB
RH6
POA
0.00 0.05 0.10 0.15 0.20 0.25 0.30
F)
POA
SPAL
RH6
PTUB
PTM2
IHYP
VGL1
CCRB
RHYP
RH3
CRBL
PALL
PTCT
TOSC
RH2
TEGM
HABN
RH1
TSPV
RH5
RH7
MTCB
0.0 0.2 0.4 0.6 0.8
G)
RH5
RH1
TOSC
RHYP
CRBL
RH2
RH7
CCRB
MTCB
VGL1
TSPV
PALL
HABN
RH3
TEGM
RH6
PTCT
PTUB
IHYP
SPAL
PTM2
POA
0.00 0.02 0.04 0.06 0.08 0.10 0.12

Figure S7: (Related to Figure 7) Functional subdivisions of thermoregulatory circuits.

A-C) For each analyzed Z-Brain region the units encoding swim-starts **(A)**, vigor **(B)** or directionality **(C)** as a fraction of the sum across these three groups. Red bars highlight the top regions with significant enrichment plotted in Figure 6A. Open bars and gray text highlight those regions that are neither significantly enriched nor significantly depleted of the given type. Blue line is the expected (average) fraction of the given type. Error-bars mark a boot-strapped 95% confidence interval around the average of possible observed fractions if the region would in fact contain the expected fraction of units. Bars for which the top is above this confidence interval are therefore significantly enriched, bars for which the top is below are significantly depleted. Note that regions in which less than 50 neurons total across the listed categories have been identified, have not been analyzed for enrichment. Region abbreviations are listed in Table S1.

D) Temporal receptive fields of all clusters. Each plot shows the influence of a change in temperature on the activity (as measured by Calcium) of a neuron within the cluster. Numbers after the colon reflect the number of neurons present in each given cluster.

E-G) Same as A)-C) but for sensory units of complexity 0 **(E)**, complexity 1 **(F)** or complexity 2 **(G)**. Red bars highlight the top regions with significant enrichment plotted in Figure 6G. Note, that since this comparison only contains stimulus related units, much fewer regions pass the minimum threshold of 50 neurons .

Table S1: Z-Brain region abbreviations

Abbreviation	Z-Brain Region Name
AC	Telencephalon - Anterior Commissure
ARPS	Rhombencephalon - Area Postrema
AVTM	Rhombencephalon - Anterior Cluster of nV Trigeminal Motorneurons
CCRB	Rhombencephalon - Corpus Cerebelli
CHYP	Diencephalon - Caudal Hypothalamus
CRBL	Rhombencephalon - Cerebellum
DTHL	Diencephalon - Dorsal Thalamus
EMGR	Rhombencephalon - Eminentia Granularis
ETHL	Diencephalon - Eminentia Thalami
GAD1	Rhombencephalon - Gad1b Stripe 1
GAD2	Rhombencephalon - Gad1b Stripe 2
GAD3	Rhombencephalon - Gad1b Stripe 3
GLY1	Rhombencephalon - Glyt2 Stripe 1
GLY2	Rhombencephalon - Glyt2 Stripe 2
GLY3	Rhombencephalon - Glyt2 Stripe 3
HABN	Diencephalon - Habenula
IHYP	Diencephalon - Intermediate Hypothalamus
INFO	Rhombencephalon - Inferior Olive
IPN	Rhombencephalon - Interpeduncular Nucleus
ISL1	Rhombencephalon - Isl1 Stripe 1
LBCC	Rhombencephalon - Lobus caudalis cerebelli
LCOE	Rhombencephalon - Locus Coeruleus
LREN	Rhombencephalon - Lateral Reticular Nucleus
MTCB	Mesencephalon - Medial Tectal Band
MVNC	Rhombencephalon - Medial Vestibular Nucleus
NMLF	Mesencephalon - NucMLF (nucleus of the medial longitudinal fascicle)
NNIV	Rhombencephalon - Noradrendergic neurons of the Interfascicular and Vagal areas
OC	Telencephalon - Optic Commissure
OCN3	Mesencephalon - Oculomotor Nucleus nIII
OCN4	Rhombencephalon - Oculomotor Nucleus nIV
OLI2	Rhombencephalon - Olig2 Stripe
PALL	Telencephalon - Pallium
PINE	Diencephalon - Pineal
PITU	Diencephalon - Pituitary
POA	Diencephalon - Preoptic Area
POC	Telencephalon - Postoptic Commissure
POCM	Diencephalon - Postoptic Commissure
PTCT	Diencephalon - Pretectum
PTM1	Diencephalon - Migrated Area of the Pretectum (M1)

Abbreviation	Z-Brain Region Name
PTM2	Diencephalon - Migrated Posterior Tubercular Area (M2)
PTUB	Diencephalon - Posterior Tuberculum
PVTM	Rhombencephalon - Posterior Cluster of nV Trigeminal Motorneurons
RCAD	Rhombencephalon - CaD
RCAV	Rhombencephalon - CaV
RH1	Rhombencephalon - Rhombomere 1
RH2	Rhombencephalon - Rhombomere 2
RH3	Rhombencephalon - Rhombomere 3
RH4	Rhombencephalon - Rhombomere 4
RH5	Rhombencephalon - Rhombomere 5
RH6	Rhombencephalon - Rhombomere 6
RH7	Rhombencephalon - Rhombomere 7
RHYP	Diencephalon - Rostral Hypothalamus
RPIN	Rhombencephalon - Raphe - Inferior
RPSP	Rhombencephalon - Raphe - Superior
SPAL	Telencephalon - Subpallium
SPOS	Telencephalon - Subpallial Otpb strip
TEGM	Mesencephalon - Tegmentum
TEM4	Telencephalon - Telencephalic Migrated Area 4 (M4)
TOLO	Mesencephalon - Torus Longitudinalis
TOLT	Diencephalon - Torus Lateralis
TOSC	Mesencephalon - Torus Semicircularis
TSPV	Mesencephalon - Tectum Stratum Periventriculare
VCRB	Rhombencephalon - Valvula Cerebelli
VGL1	Rhombencephalon - Vglut2 Stripe 1
VGL2	Rhombencephalon - Vglut2 Stripe 2
VGL3	Rhombencephalon - Vglut2 Stripe 3
VGL4	Rhombencephalon - Vglut2 Stripe 4
VII	Rhombencephalon - VII Facial Motor and octavolateralis efferent neurons
VIIP	Rhombencephalon - VII' Facial Motor and octavolateralis efferent neurons
VMT1	Rhombencephalon - Vmat2 Stripe1
VMT2	Rhombencephalon - Vmat2 Stripe2
VMT3	Rhombencephalon - Vmat2 Stripe3
VTHL	Diencephalon - Ventral Thalamus
XVMN	Rhombencephalon - X Vagus motor neuron cluster