Learnable latent embeddings for joint behavioral and neural analysis

Steffen Schneider[1 *], **Jin Hwa Lee**[1 *], **& Mackenzie Weygandt Mathis**[1 ✉]

[1]École Polytechnique Fédérale de Lausanne (EPFL), Brain Mind Institute & Neuro-X. Geneva, Switzerland

[*]co-first authors, ✉mackenzie@post.harvard.edu

Mapping behavioral actions to neural activity is a fundamental goal of neuroscience. As our ability to record large neural and behavioral data increases, there is growing interest in modeling neural dynamics during adaptive behaviors to probe neural representations. In particular, neural latent embeddings can reveal underlying correlates of behavior, yet, we lack non-linear techniques that can explicitly and flexibly leverage joint behavior and neural data. Here, we fill this gap with a novel method, CEBRA, that jointly uses behavioral and neural data in a hypothesis- or discovery-driven manner to produce consistent, high-performance latent spaces. We validate its accuracy and demonstrate our tool's utility for both calcium and electrophysiology datasets, across sensory and motor tasks, and in simple or complex behaviors across species. It allows for single and multi-session datasets to be leveraged for hypothesis testing or can be used label-free. Lastly, we show that CEBRA can be used for the mapping of space, uncovering complex kinematic features, and rapid, high-accuracy decoding of natural movies from visual cortex.

A central quest in neuroscience is the neural origin of behavior (1, 2). Yet, we are still limited in both the number of neurons and length of time we can record from behaving animals in a session. Therefore, we need new methods that can combine data across animals and sessions with minimal assumptions, and generate interpretable neural embedding spaces (1, 3). Current tools for representation learning are either linear, or if non-linear they typically rely on generative models, and they do not yield consistent embeddings across animals (or repeated runs of the algorithm). Here, we combine recent advances in non-linear disentangled representation learning and self-supervised learning to develop a new dimensionality reduction method that can be applied jointly to behavioral and neural recordings to reveal meaningful lower dimensional neural population dynamics (3–5).

From data visualization (clustering) to discovering latent spaces that explain neural variance, dimensionality reduction of behavior or neural data has been impactful in neuroscience. For example, complex 3D forelimb reaching can be reduced to only 8–12 dimensions (6, 7), and the low dimensional embeddings reveal some robust aspects of movements (i.e., PCA-based manifolds where the neural state space can easily be constrained and is stable across time (8–10)). Linear methods such as PCA are often used to increase interpretability, but this comes at the cost of performance (1). UMAP (11) and tSNE (12) are excellent non-linear methods, but they lack the ability to explicitly use time information, which is always available in neural recordings, and they are not directly as interpretable as PCA.

Non-linear methods are desirable to use for high performance decoding, but often lack identifiability: the desirable property that true model parameters can be determined, up to a known indeterminacy (13, 14). This is critical as it ensures that the learned representations are uniquely determined and thus facilitates consistency across animals and/or sessions.

There is recent evidence that label-guided VAEs could improve interpretability (5, 15, 16). Namely, by using behavioral variables, such algorithms can learn to project future behavior onto past neural activity (15), or explicitly use label-priors to shape the embedding (5). However, these methods still have restrictive explicit assumptions on the underlying statistics of the data, and they do not guarantee consistent neural embeddings across animals (5, 17, 18), which limits their generalizability as well as interpretability (and thereby affects accurate decoding across animals).

We address these open challenges with CEBRA, a new self-supervised learning algorithm for obtaining interpretable, **C**onsistent **EmB**eddings of high-dimensional **R**ecordings using **A**uxiliary variables. Our method combines ideas from non-linear independent component analysis (ICA) with contrastive learning (14, 19–21), a powerful self-supervised learning scheme, to generate latent embeddings conditioned on behavior (auxiliary variables) and/or time. CEBRA uses a novel data sampling scheme to train a neural network encoder with a contrastive optimization objective to shape the embedding space. It also can generate embeddings across multiple subjects, and cope with distribution shifts between experimental sessions, subjects, and recording modalities. Importantly, our method neither relies on data augmentation (as does SimCLR (22)), nor on a specific generative model that would limit its range of use, and can be used in a hypothesis-driven (behaviorally-guided), data-driven (time-only), or hybrid manner. We can uncover latent embeddings that are consistent and informative: we can decode behaviors with high accuracy, which we demonstrate for datasets from vision, sensorimotor, and memory systems.

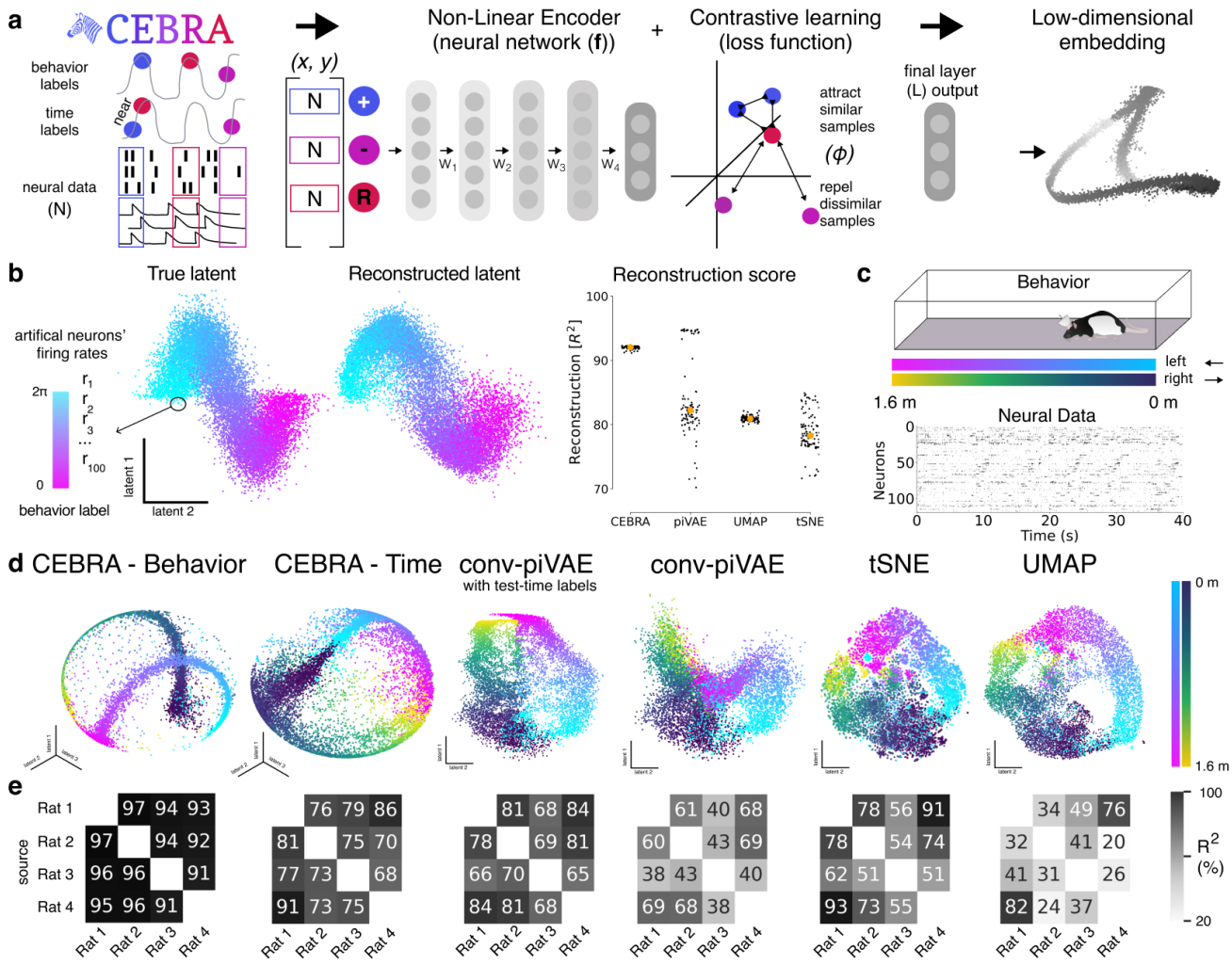


Figure 1. CEBRA for consistent and interpretable embeddings (a): CEBRA allows for self-supervised, supervised, and hybrid approaches for hypothesis-driven and discovery-driven analysis. Overview of pipeline: collect data (e.g., pairs of behavior (or time) and neural data (x,y)), determine positive and negative pairs, train CEBRA, and produce embeddings. **(b)**: Left: True 2D latent, where each point is mapped to spiking rate of 100 neurons. (Middle): CEBRA embedding after linear regression to the true latent. Right: Reconstruction score is R^2 of linear regression between the true latent and resulting embedding from each method. The "behavior label" is a 1D random variable sampled from uniform distribution of [0, 2π] that is assigned to each time bin of synthetic neural data, visualized by the color map. The orange line is the median,and each black dot is an individual run (n=100). CEBRA-Behavior shows significantly higher reconstruction score compare to pi-VAE, tSNE and UMAP (one-way ANOVA, F(3, 396)=278.31, p=3.95e-97 with post hoc Tukey HSD p<0.001). **(c)**: Rat hippocampus data from (23). Electrophysiology data collected during a task where the animal transverse a 1.6m linear track "leftwards" or "rightwards". **(d)**: We benchmarked CEBRA against conv-pi-VAE (both with labels and without (self-supervised mode)), tSNE, and unsupervised UMAP. Note, for performance against the original pi-VAE see Extended Data Fig. S1. We plot the 3 latents (note, all CEBRA embedding figures show the first 3 latents).The dimensionality (D) of the latent space is set to the minimum and equivalent dimension per method (3D for CEBRA and 2D for others) to fairly compare. Note, higher dimensions for CEBRA can give higher consistency values (see Extended Data Fig. S7). **(e)**: Correlation matrices depict the R^2 after fitting a linear model between behavior-aligned embeddings of two animals, one as the target one as the source (mean, n=10 runs). Parameters were picked by optimizing average run consistency across rats.

Results

Joint behavioral and neural embeddings.

We propose a framework for jointly trained latent embeddings. CEBRA leverages user-defined labels (hypothesis-driven), or time-only labels (discovery-driven; Fig. 1a, Suppl. Note 1) to obtain consistent embeddings of neural activity that can be used for both visualization of data and downstream tasks like decoding. Specifically, it is an instantiation of non-linear ICA based on contrastive learning (14). Contrastive learning is a technique that leverages contrasting samples (positive and negative) against each other to find attributes in common and those that separate them. We can use discrete and continuous variables and/or time to shape the distribution of positive and negative pairs, and then use a non-linear encoder (here, a convolutional neural network (CNN), but can be another type of model) trained with a novel contrastive learning objective. The encoder features form a low-dimensional embedding of the data (Fig. 1a). Generating consistent embeddings is highly desirable and closely linked to identifiability in non-linear ICA (24). Theoretical work has shown that using contrastive learning with auxiliary variables is identifiable for bijective neural networks using a noise contrastive estimation (NCE) loss (14), and that with an InfoNCE loss this bijectivity assumption can sometimes be removed (25) (see also Suppl. Note 2). InfoNCE min-

imization can be viewed as a classification problem where given a reference sample, the correct positive pair needs to be distinguished from multiple negative pairs.

CEBRA optimizes a neural network $\mathbf{f}$ that maps neural activity to an embedding space of a defined dimension (Fig. 1a). Pairs of data $(\mathbf{x}, \mathbf{y})$ are mapped to this embedding space, and then compared with a similarity measure $\phi(\cdot,\cdot)$. Abbreviating this process with $\psi(\mathbf{x},\mathbf{y}) = \phi(\mathbf{f}(\mathbf{x}),\mathbf{f}(\mathbf{y}))/\tau$ with a temperature hyperparameter τ, the full criterion to optimize is

$$\mathop{\mathbb{E}}_{\substack{\mathbf{x}\sim p(\mathbf{x}),\ \mathbf{y}_+\sim p(\mathbf{y}|\mathbf{x}) \\ \mathbf{y}_1,\dots,\mathbf{y}_n\sim q(\mathbf{y}|\mathbf{x})}} \left[-\psi(\mathbf{x},\mathbf{y}_+) + \log \sum_{i=1}^{n} e^{\psi(\mathbf{x},\mathbf{y}_i)} \right],$$

which, depending on the dataset size, can be optimized with algorithms for either batch or stochastic gradient descent.

In contrast to other contrastive learning algorithms, the positive pair distribution p and the negative pair distribution q can be systematically designed and allows the use of time, behavior, and other auxiliary information to shape the geometry of the embedding space. If only discrete labels are used, this training scheme is conceptually similar to supervised contrastive learning (21).

CEBRA can leverage continuous behavioral (kinematics, actions) as well as other discrete variables (trial ID, rewards, brain-area ID, etc.). Additionally, user-defined information about desired invariances in the embedding is used (across animals, sessions, etc.), allowing flexible ways of analyzing data. We group this information into task-irrelevant and task-relevant variables, and these can be leveraged in different contexts. For example, to investigate trial-to-trial variability or learning across trials, information like a trial ID would be considered a task-relevant variable. On the contrary, if we aim to build a robust brain machine interface that should be invariant to such short-term changes, we would include trial information as a task-irrelevant variable and obtain an embedding space which no longer carries this information. Crucially, this allows for inferring latent embeddings without explicitly modeling the data generating process (as done in pi-VAE (5) and LFADS (17)). Omitting the generative model and replacing it by a contrastive learning algorithm facilitates broader applicability without modifications.

Robust and decodable latent embeddings.

We first demonstrate that CEBRA significantly outperforms tSNE, UMAP, and pi-VAE (the latter was shown to outperform PCA, LFADS, demixed-PCA, and pfDS (5)) at reconstructing ground truth synthetic data (one-way ANOVA, F(3, 396)=278.31, p=3.95e-97; Fig. 1b, Extended Data Fig. S1a, b).

We then turned to a hippocampus dataset that was used to benchmark neural embedding algorithms (5, 23) (Extended Data Fig. S1c, Suppl. Note 1). To note, we first significantly improved pi-VAE by adding a CNN thereby allowing this model to leverage multiple time steps, and used this for further benchmarking (Extended Data Fig. S1d-e). To test our methods, we first consider the correlation of the resulting embedding space across subjects (does it produce similar latent spaces?), and the correlation across repeated runs of the algorithm (how consistent are the results?). We found that CEBRA significantly outperformed other algorithms at producing consistent embeddings, and it produced visually informative embeddings (Fig. 1c-e, Extended Data Figs. S2, S3; for each embedding a single point represents the neural population activity over a specified time bin).

When using CEBRA-Behavior the correlation of the resulting embedding space across subjects is significantly higher compared to conv-pi-VAE with, or without test-time labels (one-way ANOVA F(5, 67)=28, p=3.4×10^{-15}, Table 1; Fig. 1d, e)—note, CEBRA does not require test-time labels. Qualitatively, it can be appreciated that both CEBRA-Behavior and -Time have similar output embeddings, while the latents from conv-pi-VAE with label priors or without labels are not consistent: namely, conv-pi-VAE without label priors results in a more entangled latent, suggesting that the label prior strongly shapes the output embedding structure of conv-pi-VAE. We also considered correlations across repeated runs of the algorithm and found higher consistency and lower variability with CEBRA (Fig. 1b, Extended Data Fig. S4).

Hypothesis- and discovery-driven analyses.

One of the advantages of CEBRA is its flexibility, limited assumptions, and ability to test hypotheses. For the hippocampus, one can hypothesize that these neurons represent space (26, 27), and therefore the behavioral label could be position, or velocity (Figure 2a). Conversely, for the sake of argument, we could have an alternative hypothesis; i.e., hippocampus does not map space, just the direction of travel, or some other feature. Using the same model, but hypothesis-free and using time for selecting the contrastive pairs is also possible, and/or a hybrid thereof (Fig. 2a).

We trained hypothesis-guided, time-only, or hybrid models across a range of input dimensions and embedded the neural latents into a 3D space for visualization. Qualitatively, we find that position-based model produces a highly smooth embedding that reveals the position of the animal—namely, there is a continuous "loop" of neural latent activity around the track (Fig. 2b). This is consistent with what is known about the hippocampus (23) and in particular reveals the topology of the linear track with direction specificity. Whereas shuffling the labels, which breaks the correlation between neural activity and direction and position, produces an unstructured embedding (Fig. 2b).

CEBRA-Time produces an embedding that more closely resembles that of position (Fig. 2b). This also suggests that time contrastive learning captured the major latent space structure, independent of any label input, reinforcing that CEBRA can serve both discovery and hypothesis-driven questions (and running both variants can be informative). The hybrid design, whose goal is to disentangle the latent to subspaces that are relevant to the given behavioral and the resid-

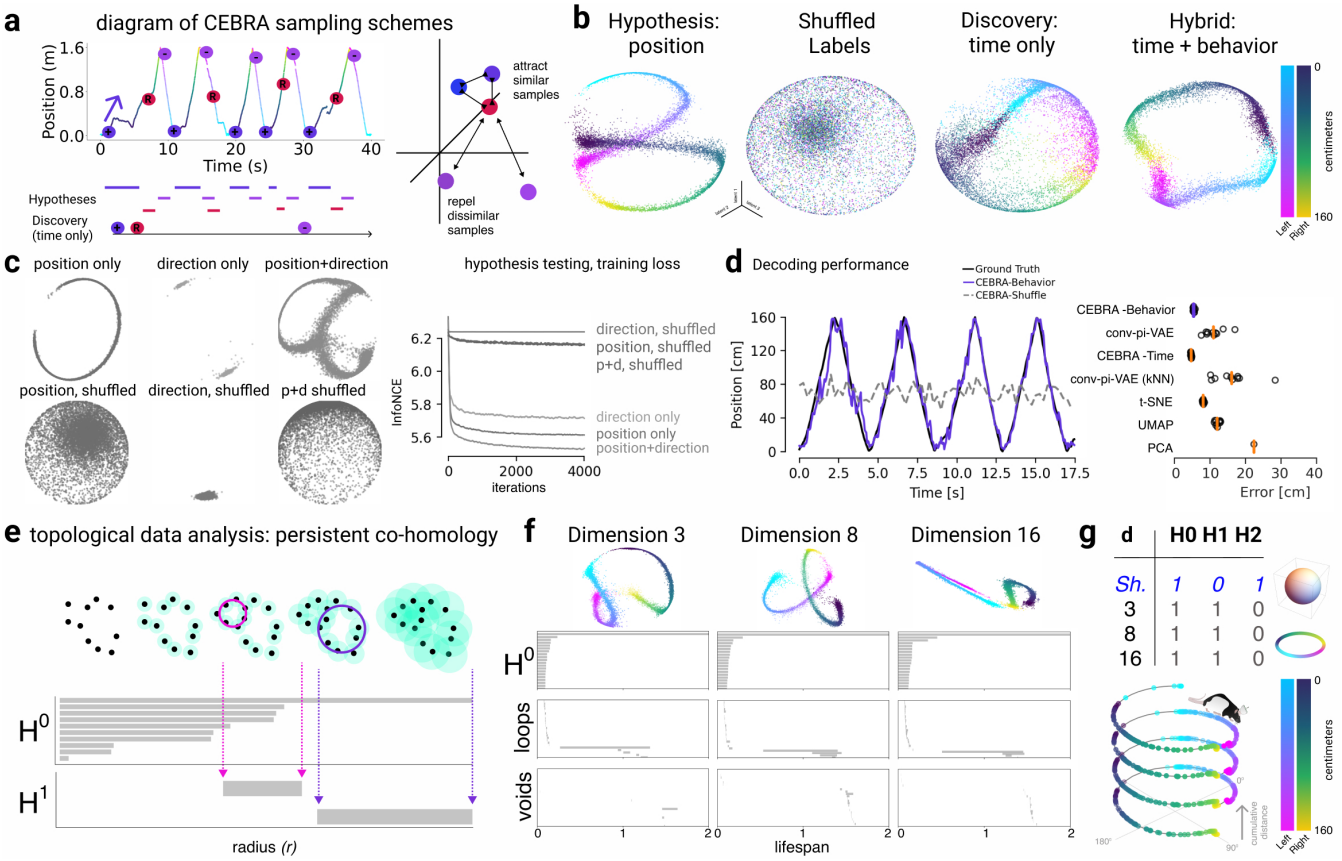


Figure 2. Hypothesis-driven and discovery-driven analysis with CEBRA (a): CEBRA can be used in three modes: hypothesis-driven, discovery-driven, or in a hybrid mode, which allows for weaker priors on the latent embedding. **(b)**: CEBRA with position-hypothesis derived embedding, shuffled (erroneous), time-only, and Time+Behavior (hybrid; here, a 5D space was used, where first 3D is guided by both behavior+time, and last 2D is guided only by time, and the first 3 latents are plotted). **(c)**: Embeddings with position-only, direction-only, and shuffled position-only, direction-only for hypothesis testing. The loss function can be used as a metric for embedding quality. **(d)**: We utilized the hypothesis-driven (position+direction) or the shuffle (erroneous) to decode the position of the rat, which produces a large difference in decoding performance: position+direction R^2 is 73.35% vs. -49.90% shuffled and median absolute error 5.8 cm vs 44.7 cm. Purple line is decoding from the 32-dimensional hypothesis-based latent space, dashed line is shuffled. Right is the performance across additional methods (The orange line indicates the median of the individual runs (n=10) that are indicated by black circles. Each run is averaged over 3 splits of the dataset). **(e)**: Schematic of how persistent co-homology is computed. Each data point is thickened to a ball of gradually expanding radius r, while tracking birth and death of "cycles" in each dimension (H^0 counts number of connected components or 0-dim cycles, H^1 counts the number of loops (1-dim cycles), H^2 counts the number of voids (2-dim cycles)). The prominent lifespans, indicated as pink and purple arrows, are considered to determine Betti numbers. **(f)**: Visualization of the neural embeddings computed with different input dimensions, and the related persistent co-homology lifespan diagrams below. **(g)**: Betti numbers from shuffled embeddings (Sh.) and across increasing dimensions (d) of CEBRA, and the topology preserving circular coordinates using the first co-cycle from persistent co-homology analysis (see Methods).

ual temporal variance and noise, showed a similarly structured embedding space as behavior (Fig. 2b).

To quantify how CEBRA can disentangle which variable had the largest influence on the embedding, we tested for encoding position, direction, and combinations thereof (Fig. 2c). We find that position plus direction is the most informative label (28) (Fig. 2c, and Extended Data Fig. S5a-d). This is evident in the embedding and the value of the loss function upon convergence, which serves as an additional "goodness of fit" metric to select the best labels; i.e., which label(s) produce the lowest loss at the same point in training (Extended Data Fig. S5e). Note that erroneous (shuffled) labels converge to considerably higher loss values.

To measure performance we consider how well can we decode behavior from the embeddings. As an additional baseline we performed linear dimensionality reduction with PCA. We used a k-nearest-neighbor (kNN) decoder for position and direction and measured the reconstruction error. We find CEBRA-Behavior has significantly better decoding performance (Fig. 2d, and Suppl. Video 1), compared to pi-VAE and our conv-pi-VAE (one-way ANOVA F=131, p=3.6×10^{-24}), and CEBRA-Time compared to unsupervised methods (tSNE, UMAP, PCA; one-way ANOVA F=12 091 p=6.95×10^{-42}; see also Table 2). Zhou and Wei (5) reported a median absolute decoding error of 12 cm error, while we can achieve ≈5 cm (Fig. 2d). CEBRA therefore allows for high performance decoding while ensuring consistent embeddings.

Co-homology as a metric for robust embeddings.

CEBRA can be trained across a range of dimensions and models can be selected based on decoding, goodness of fit, and consistency. Yet, we also sought to find a principled approach to verify the robustness of embeddings, which might yield insight into neural computations (30, 31) (Fig. 2e). We used algebraic topology to measure the persistent co-

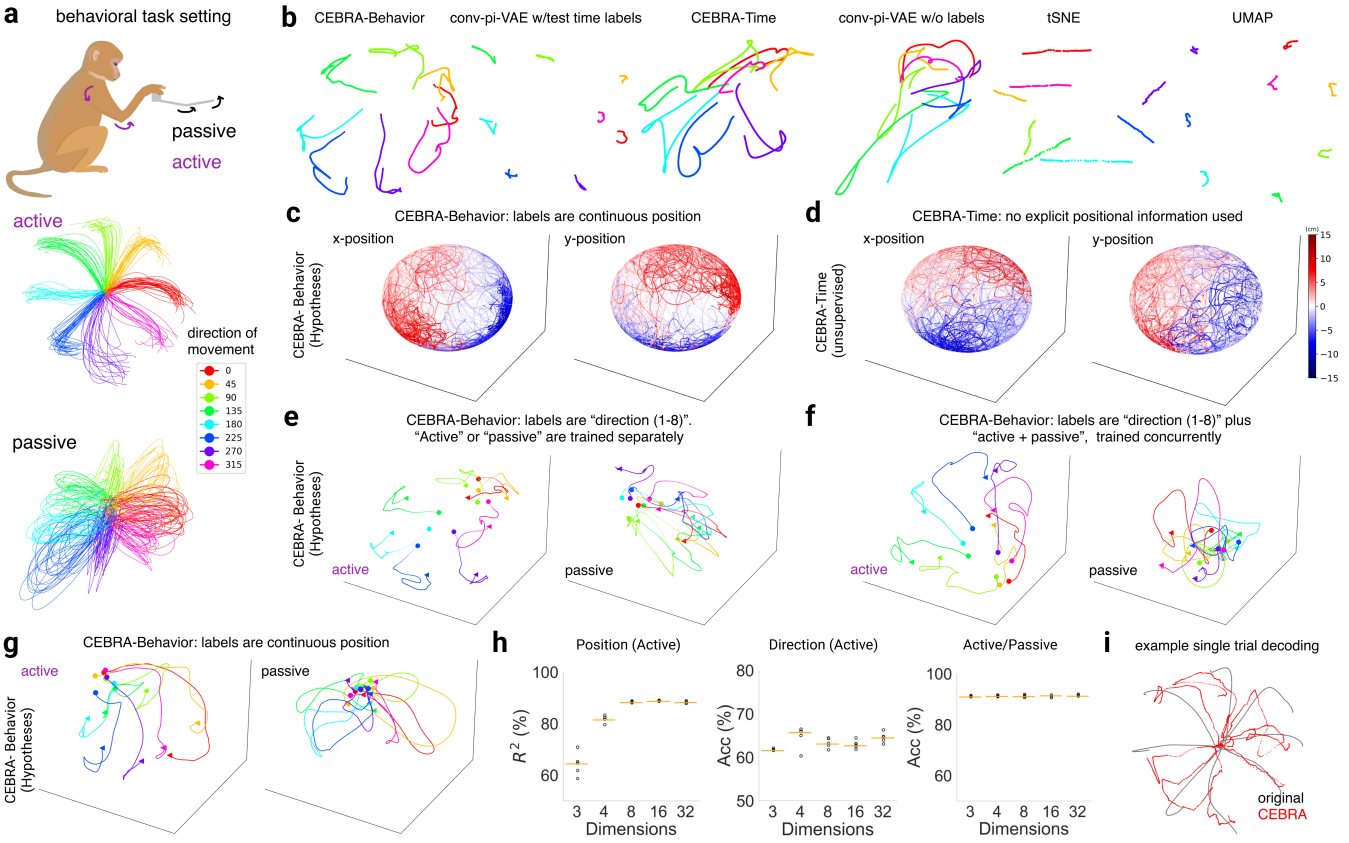


Figure 3. Forelimb movement behavior in a primate (a): Behavioral setup: monkey makes either active movements in 8 directions with the manipulandum, or the arm is passively moved via the manipulandum (real behavioral trajectories shown, with cartoon depicting the task setup). Behavior and neural recordings are from area 2 of the primary somatosensory cortex from Chowdhury et al. (29). **(b)**: Comparison of embeddings of active trials generated with CEBRA-Behavior, CEBRA-Time, conv-pi-VAE variants, tSNE, and UMAP. The embeddings of trials (n=364) of each direction are post-hoc averaged. **(c)**: CEBRA-Behavior trained with x,y position of the hand. Left panel is color-coded to x position and right panel is color-coded to y position, as in **d**. **(d)**: CEBRA-Time without any external behavior variables. As in **c**, left and right are color-coded to x and y position, respectively. **(e)**: Left, CEBRA-Behavior embedding trained with a 4D latent space, with discrete target direction as behavior labels, trained and plotted separately for active and passive trials. **(f)**: Left, CEBRA-Behavior embedding trained with a 4D latent space, with discrete target direction and active and passive trials as behavior labels, plotted separately, active vs. passive trials. **(g)**: CEBRA-Behavior embedding trained with a 4D latent space using active and passive trials with continuous (x,y) position as behavior labels, plotted separately, active vs. passive trials. The trajectory of each direction is averaged across trials (n=18–30 each, per directions) over time. Each trajectory represents 600ms from -100ms before the start of the movement. **(h)**: Left to right: Decoding performance of: position using CEBRA-Behavior trained with x,y position (active trials); target direction using CEBRA-Behavior trained with target direction (active trials); or active vs. passive accuracy using CEBRA-Behavior trained with both active and passive movements. For each case, we trained and evaluated 5 seeds represented by black dot and the orange line represents median. **(i)**: Decoded trajectory of hand position using CEBRA-Behavior trained on active trial with x,y position of hand. Grey line is true trajectory and red line is decoded trajectory.

homology, for comparing if learned latent spaces are equivalent. While it is not required to project embeddings onto a sphere, this has the advantage that there are default Betti numbers (for a d-dimensional uniform embedding, $H^0 = 1, H^1 = 0, \cdots, H^{d-1} = 1$, i.e., 1,0,1 for the 2-sphere). We used the distance from the unity line (and thresholded based on a computed null shuffled distribution in Births vs. Deaths to compute Betti numbers; Extended Data Fig. S6). Using CEBRA-Behavior or -Time we find a ring topology (1,1,0; Fig. 2f), as one would expect from a linear track for place cells. We then computed the Eilenberg-MacLane coordinates for the identified co-cycle (H1) for each model (32, 33)—this allowed us to map each time-point to topology-preserving coordinates—and indeed we find that the ring topology for the CEBRA models matches space (position) across dimensions (Fig. 2g, Extended Data Fig. S6). Note, this topology differs from (1,0,1); i.e., Betti numbers for a uniformly covered sphere, which in our setting would indicate a random embedding as found by shuffling (Fig. 2g).

Multi-session, multi-animal CEBRA.

CEBRA can also be used to jointly train across sessions and different animals, which can be highly advantageous when there is limited access to simultaneously recorded neurons, or when looking for animal-invariant features in the neural data. We trained CEBRA across animals within each multi-animal dataset and find this joint embedding allows for even more consistent embeddings across subjects (Extended Data Fig. S7a-c; one-sided, paired T-tests, Allen data: (-5.80), $p=5.99 \times 10^{-5}$; Hippocampus: (-2.22), p=0.024).

While consistency increased, it is not *a priori* clear that decoding from “pseudo-subjects” would be equally good, as there could be session or animal specific information that is lost in pseudo-decoding (as decoding is usually performed within session). Alternatively, if this joint latent space was

as high-performance as the single subject, this would suggest that CEBRA is able to produce robust latent spaces across subjects. Indeed, we find no loss in decoding performance (Extended Data Fig. S7c).

It is also possible to rapidly decode from a new session that is *unseen* during training, which is an attractive setting for brain machine interface deployment. We show that by pre-training on a subset of the subjects, we can apply and rapidly adapt CEBRA-Behavior on unseen data (i.e., it runs at 50–100 steps/second, and positional decoding error already decreased by 10 cm after adapting the pretrained network for one step). Lastly, we can achieve a lower error more rapidly compared to training fully on the unseen individual (Extended Data Fig. S7d). Collectively, this shows that CEBRA can rapidly produce high-performance, consistent and robust latent spaces.

Discovering latent dynamics during a motor task.

We next consider an eight direction "center-out" reaching task paired with electrophysiology recordings in somatosensory cortex (S1) of a primate (29) (Fig. 3a). The monkey performed active movements and in a subset of trials experienced randomized bumps that caused a passive limb movement. CEBRA produced highly informative visualisations of the data compared to other methods (Fig. 3b), and CEBRA-Behavior can be used in order to test encoding properties of S1. Using position or time information showed embeddings with clear positional encoding (Fig. 3c, d, and Extended Data Fig. S8a-c).

To then test how directional information and active vs. passive movements influence population dynamics in S1 (29, 34, 35), we trained embedding spaces with directional information and then either separated the trials into active and passive for training (Fig. 3e), or trained jointly and post-hoc plotted separately (Fig. 3f). We find striking similarities that suggest active vs. passive strongly influences the neural latent space: the embeddings for active trials show a clear start and stop, while for passive trials it shows a continuous trajectory through the embedding, independently of how they are trained. This finding is confirmed in embeddings that used only the *continuous* position of the end-effector as the behavioral label (Fig. 3g). Notably, direction is a less prominent feature (Fig. 3g), although they are entangled parameters in this task.

Next, since position and active or passive trial type appear robust in the embeddings, we further explored the decodability of the embeddings. Both position and trial type were readily decodable from 8D+ embeddings with a kNN decoder trained on position-only, but directional information was not as decodable (Fig. 3h). Here too the loss function is informative for hypothesis testing (Extended Data Fig. S8d-f). Notably, we could recover the hand trajectory with an R^2 of 88% (con-

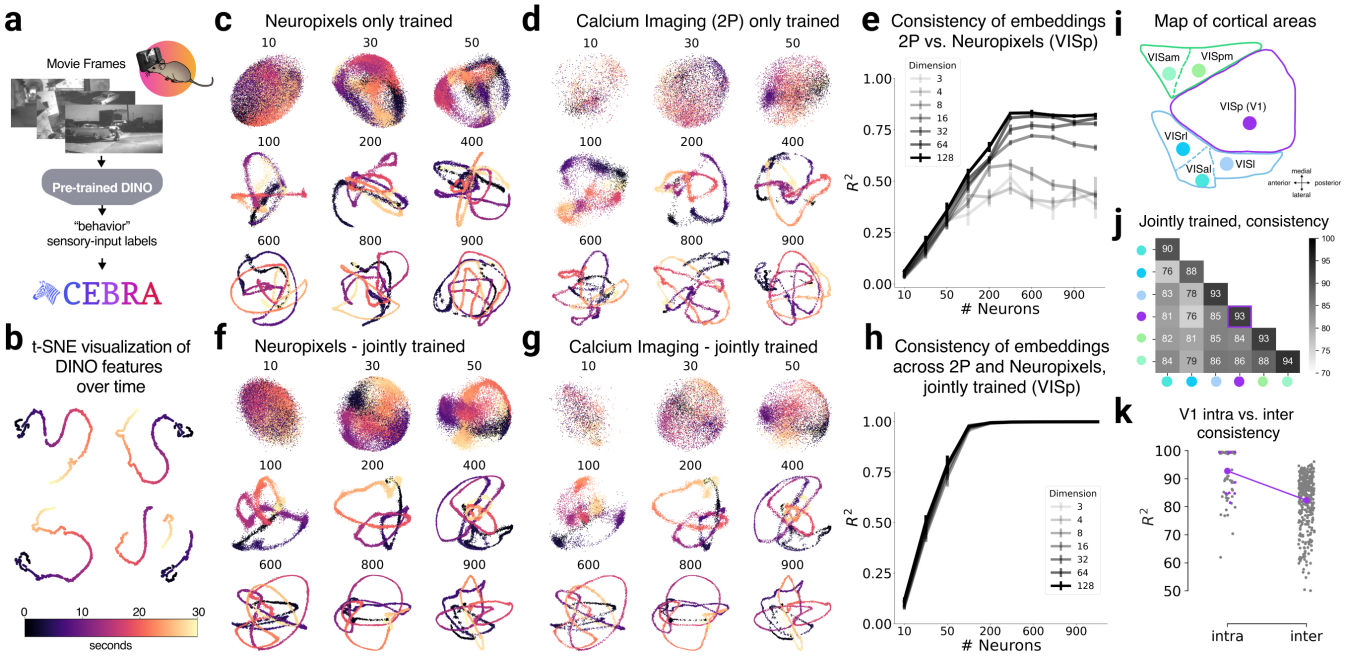


Figure 4. Spikes and calcium signaling reveal similar CEBRA embeddings (a): CEBRA-Behavior can use frame-by-frame video feature as a label of sensory input to extract neural latent space of visual cortex of mice watching a movie. **(b)**: tSNE visualization of the DINO features of the movie frames from four different DINO configurations (latent size, model size) commonly show continuous evolution of the movie frames over time. **(c, d)**: Visualization of trained 8D latent CEBRA-Behavior embeddings with Neuropixels data or calcium imaging, respectively. The numbers on top of each embedding is the number of neurons subsampled from the multi-session concatenated dataset. Color map is the same as in **b**. **(e)**: Linear consistency between embeddings trained with either calcium imaging data or Neuropixels data. **(f, g)**: Visualization of CEBRA-Behavior embedding (8D) trained with Neuropixels and calcium imaging, jointly. Color map is the same as in **b**. **(h)**: Linear consistency between embeddings of calcium imaging and Neuropixels which were trained jointly using a multi-session CEBRA model. **(i)**: Diagram of mouse primary visual cortex (V1, VIsp), PPC (VIsrl) and higher visual areas. **(j)**: CEBRA-Behavior 32D model jointly trained with 2P+NP with 400 neurons then consistency measured within or across areas (2P vs. NP) across 2 unique sets of disjoint neurons for 3 seeds and averaged. **(k)**: Models trained as in **h**, with intra-V1 consistency measurement vs. all inter-area vs. V1 comparison. Purple dots indicate mean of V1 intra-V1 consistency (across n=12 runs) and inter-V1 consistency (n=60). Intra-V1 consistency is significantly higher than inter-area consistency (Welch's t-test, T(19,53)=4.55, p=0.00019).

catenated across 26 held-out test trials, Fig. 3i) using a 16D CEBRA-Behavior model trained on position (Fig. 3i). For comparison, a L1 regression using all neurons achieved R^2 74%, and 16D conv-pi-VAE achieved R^2 82%, making our approach state-of-the-art on this data.

Consistent embeddings across modalities.

CEBRA is agnostic to the recording modality of neural data. But do different modalities produce similar latent embeddings? Understanding the relationship of calcium signaling and electrophysiology is a debated topic, yet an underlying assumption is that they inherently encode related, yet not identical, information. Although there are a wealth of excellent tools aimed at inferring spike trains from calcium data, currently the pseudo-R^2 of algorithms on paired spiking and calcium data tops out at around 0.6 (36). Nonetheless, it is clear that recording with either modality has lead to similar global conclusions—for example, grid cells can be uncovered in spiking or calcium signals (33, 37), reward prediction errors can be found in dopamine neurons across species and recording modalities (38–40), and visual cortex shows orientation tuning across species and modalities (41–43).

We aimed to formally study whether CEBRA could capture the same neural population dynamics either from spikes or calcium imaging. We utilized a dataset from the Allen Brain Observatory where mice passively watched three movies repeatedly. We focused on paired data from 10 repeats of "Natural Movie 1" where neural data were recorded with either Neuropixels probes or via calcium imaging with a 2-photon (2P) microscope (from separate mice) (44, 45). Note, the data we considered thus far have goal-driven actions of the animals (such as running down a linear track or reaching to targets), yet this visual cortex dataset is collected during passive viewing (Fig. 4a).

We used the movie features as "behavior" labels by extracting the high-level visual features from the movie on a frame-by-frame basis using DINO, a powerful vision transformer (46). Those were then used to sample the neural data with feature-labels (Fig. 4b). Next, we used Neuropixels data or calcium (2P) data (each with multi-session training) in order to generate (from 8D to 128D) latent spaces from varying number of neurons recorded from V1 (Fig. 4c, d). The visualization of CEBRA-Behavior showed trajectories that smoothly capture the video with either modality with an increasing number of neurons. This is reflected quantitatively in the consistency metric (Fig. 4e). Strikingly, CEBRA-Time nicely captured the 10 repeats of the movie (Extended Data Fig. S9). This result demonstrates that there is a highly consistent latent space independent of the recording method.

Next, we stacked the neurons from different mice and modalities and then sampled random subsets of V1 neurons to construct a pseudo-mouse. We did not find that joint training lowered consistency within modality (Extended Data Fig. S10a, b), and overall we found considerable improvement in consistency with joint training (Fig. 4f-h).

Using CEBRA-Behavior or -Time we trained models on four higher visual areas (HVAs) and one posterior parietal cortex (PPC) area and measured the consistency with and without joint training, and within or across areas. Our results show that with joint training intra-area consistency is higher vs. other areas (Fig. 4i-k), suggesting that with CEBRA we are not removing biological differences across areas (that have known differences in decodability and feature representations (47, 48)). Moreover, we test within modality and find a similar effect with CEBRA-Behavior or -Time within recording modality (Extended Data Fig. S10c-f).

Decoding of natural movies from visual cortex.

We performed V1 decoding analysis using CEBRA models that are either joint-modality trained, single-modality trained, or with a baseline population vector then paired with a simple kNN or naive Bayes decoder. We aimed to see if we could decode on a frame-by-frame basis the natural movie the mice watched. We used the last movie repeat as a held-out test set and nine repeats as the training set. We could achieve greater than 95% decoding accuracy, which is significantly better than the baseline decoding methods (naive Bayes or kNN) for Neuropixels recordings, and joint training CEBRA outperformed Neuropixels-only CEBRA based training (single frame: one-way ANOVA, F(3,197)=5.88, p=0.0007, Tables 3, 4, 5, Fig. 5a-d, Extended Data Fig. S10g, h). Accuracy was defined as the fraction of correct frames within a 1-second window or by the correct scene being identified. Frame-by-frame results also showed reduced frame ID errors (one-way ANOVA, F(3,16)=20.22, p=1.09×10^{-5}, n=1000 neurons, Table 6) which can be appreciated in Fig. 5e, f, Extended Data Fig. S10i, and Suppl. Video 2. The DINO features themselves did not drive performance, as shuffling the features showed poor decoding (Extended Data Fig. S10j).

Lastly, we tested decoding from other HVAs and PPC using DINO features. Overall, decoding from V1 had the highest performance, and PPC (VISrl) the lowest (Fig. 5g, Extended Data Fig. S10k). Given the high decoding performance of CEBRA, we tested if there was a particular V1 layer that was most informative. We leveraged CEBRA-Behavior by training models on each category and find that layer 2/3 and layer 5/6 show significantly higher decoding performance compared to layer 4 (one-way ANOVA, F(2,12)=9.88, p=0.003; Fig. 5h). Given the known cortical connectivity, this suggests that the non-thalamic input layers make frame information more explicit, perhaps via feedback or predictive processing.

Discussion

Mapping neural activity to behavioral outputs is one of the fundamental quests of neuroscience. Here, we present CEBRA, a new dimensionality reduction method to explicitly leverage behavior or time in order to discover latent neural embeddings. We find these embeddings provide high decoding performance across a broad spectrum of behaviors—from positional decoding in hippocampus to reconstruction of natural movies from visual cortex in the mouse. CEBRA pro-

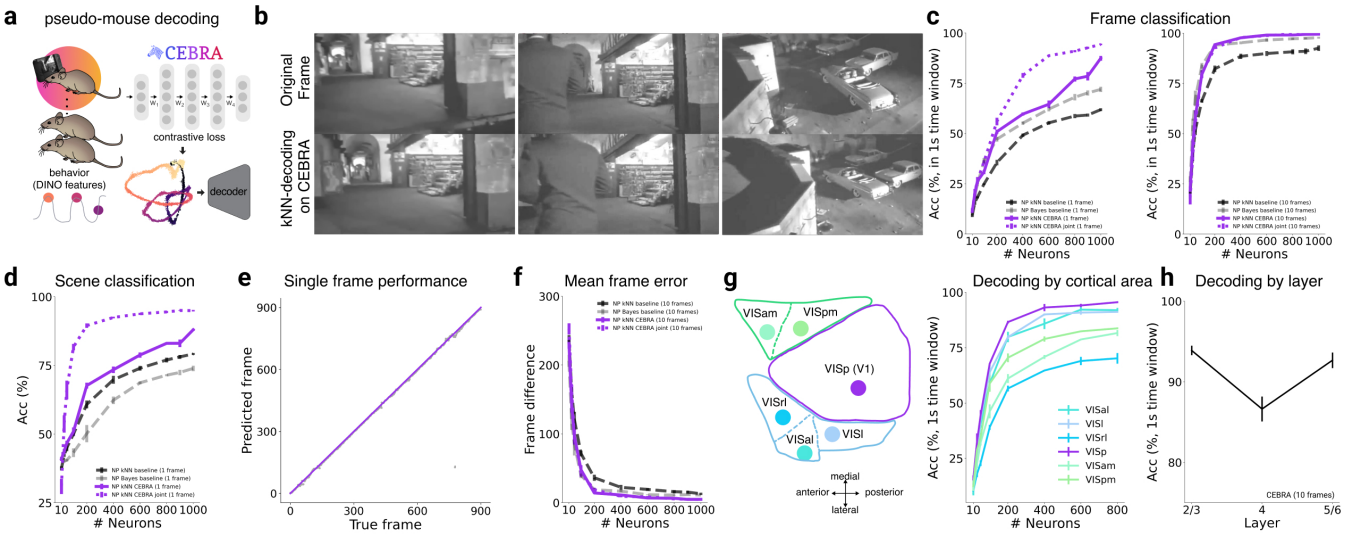


Figure 5. Decoding of natural movie features from mouse visual cortical areas. **(a)**: Schematic of the CEBRA encoder and kNN (or naive Bayes) decoder. **(b)**: Examples of original frames (top row) and frames decoded from CEBRA embedding of V1 calcium recording using kNN decoding (bottom row). The last repeat among 10 repeats was used as the held-out test. **(c)**: Decoding accuracy measured by considering a predicted frame being within 1 sec to the true frame as a correct prediction using CEBRA (NP only), jointly trained (2P+NP), or a baseline population-vector plus kNN or naive Bayes decoder using either a 1 frame (33 ms) receptive field or 10 frames (330 ms); results shown for Neuropixels dataset (V1 data). **(d)**: Decoding accuracy measured by the correct scene prediction using either CEBRA (NP only), jointly trained (2P+NP), or baseline population-vector plus kNN or Bayes decoder using a 1 frame (33 ms) receptive field (V1 data). **(e)**: Single frame ground truth frame ID vs predicted frame ID for Neuropixels using a CEBRA-Behavior model trained with a 330 ms receptive field (1,000 V1 neurons across mice used). **(f)**: The mean absolute error of the correct frame index. Shown for baseline and CEBRA models as computed in c, d, e. **(g)**: Diagram of the cortical areas considered, and decoding performance from CEBRA (NP only), 10 frame receptive field. **(h)**: V1 decoding performance vs. layer category using 900 neurons with a 330 ms receptive field CEBRA-Behavior model.

duces both consistent embeddings across subjects (thus revealing common structure) and can find the dimensionality of neural spaces that are topologically robust. While there remains a gap in understanding how these latent spaces map to neural-level computations, we believe this tool provides an advance in our ability to map behavior to neural populations.

Contrastive learning is highly attractive to use in this problem setting of using so-called auxiliary variables (14, 21). Recent work to develop more robust non-linear ICA has also shown that the InfoNCE loss encodes useful inductive biases (25). The unique property of CEBRA is the extension of the standard InfoNCE objective by introducing a variety of different *sampling strategies* tuned for usage of the algorithm in the experimental sciences, and for analysis of time series datasets. In contrast to other usages of contrastive learning (22), CEBRA does not rely on data augmentation techniques (that need to be designed specifically for a particular dataset, potentially using domain knowledge), and is still flexible and easy to adapt to different data processing needs.

Dimensionality reduction is often tightly linked to data visualization, and here we make an empirical argument that ultimately this is only useful when you are getting consistent results, and discovering robust features. Unsupervised tSNE and UMAP are examples of algorithms widely used in life sciences for discovery-based analysis. However, they do not leverage time, and for neural recordings, this is always available and can be used. Even more critical is that concatenating data from different animals can lead to shifted clusters with tSNE or UMAP due to inherent small changes across animals or in how the data was collected. CEBRA allows the user to remove this unwanted variance and discover robust latents that are invariant to animal ID, sessions, or any-other-user-defined nuisance variable. Collectively, we believe CEBRA will become a complement to (or replacement for) these methods such that, at minimum, the structure of time in the neural code is leveraged, and robustness is prioritized.

CEBRA is highly versatile: it can be used for supervised and self-supervised analysis and thereby directly allows for hypothesis- and discovery-driven science (Fig. 2). For example, our multi-session and multi-animal training allows for domain generalization and mitigation of batch effects common in biological data (i.e., constant distribution shifts that appear between recording days or sessions due to static changes in the experimental setup, acquisition method, etc.). It also allows for exploratory data analysis of time series data only, and/or data paired with a rich variety of context variables that can be used to do hypothesis-driven decoding (e.g., recordings of other—potentially confounding—signals, such as pose estimation, EMG signals, etc.), where, for example, testing dependencies between variables, or difference of experimental conditions or recording setups is of interest. We demonstrate this feature by using both continuous labels (such as position from the hippocampus task setting, Fig. 1), or discrete labels, such as "active" and "passive" in the monkey reaching dataset (Fig. 3). We also show that it does not require kinematic data, as any "behavior" labels are usable, such as DINO-based features from a natural movie which we show can be powerfully used to decode on a frame-by-frame basis images from the visual cortex of mice (Figs. 4, 5).

We demonstrate the scientific utility of CEBRA by exploring datasets collected from visual areas of mice while they passively observe a natural movie. We find that we can decode frames with greater than 95% accuracy from a "pseudo-mouse" model trained from both Neuropixels and 2P data across animals. To achieve this result, we first showed that CEBRA outperforms classical algorithms such as UMAP and tSNE and state-of-the-art neural denoising/decoding algorithm, pi-VAE. In the course of us attempting to fairly benchmark our method we incidentally improved pi-VAE (Extended Data S1). Nonetheless, we show CEBRA can significantly outperform our modified conv-pi-VAE in consistency, and decodability. Secondly, we showed that we could jointly train across animals, a feature that is not present in other methods benchmarked here (but see (17)), to generate more robust (consistent) latent spaces.

Pretrained CEBRA models can be used for decoding in new animals within tens of steps (milliseconds); we can thereby get equal or better performance compared to training on the unseen animal alone. Considering the fact that time efficiency is a highly relevant factor especially in brain machine interface applications, it is worthwhile to note that CEBRA provides much faster training compared to pi-VAE where the sampling method to approximate the test label is very time-costly. We believe our approach will be crucial for real-time, adaptive decoding.

Data Availability.

Hippocampus dataset: https://crcns.org/data-sets/hc/hc-11/about-hc-11 and we used the preprocessing script from https://github.com/zhd96/pi-vae/blob/main/code/rat_preprocess_data.py. Primate dataset: https://gui.dandiarchive.org/#/dandiset/000127. Allen Institute dataset: Neuropixels data are at https://allensdk.readthedocs.io/en/latest/visual_coding_neuropixels.html. The pre-processed 2P recordings are available at https://github.com/zivlab/visual_drift/tree/main/data. As examples with CEBRA, packaged datasets are available at https://github.com/AdaptiveMotorControlLab/CEBRA.

Code Availability.

Code: https://github.com/AdaptiveMotorControlLab/CEBRA. Documentation: https://cebra.ai/docs/. Code (and data) to reproduce the figures: https://github.com/AdaptiveMotorControlLab/CEBRA-figures. All other requests should be made to the corresponding author.

References

1. Anne E. Urai, Brent Doiron, Andrew Michael Leifer, and Anne K. Churchland. Large-scale neural recordings call for new insights to link brain and behavior. *Nature Neuroscience*, 25:11–19, 2022.
2. John W. Krakauer, Asif A. Ghazanfar, Alex Gomez-Marin, Malcolm A. MacIver, and David Poeppel. Neuroscience needs behavior: Correcting a reductionist bias. *Neuron*, 93:480–490, 2017.
3. Mehrdad Jazayeri and Srdjan Ostojic. Interpreting neural computations by examining intrinsic and embedding dimensionality of neural activity. *Current Opinion in Neurobiology*, 70:113–120, 2021.
4. Mark D Humphries. Strong and weak principles of neural dimension reduction, 2021.
5. Ding Zhou and Xue-Xin Wei. Learning identifiable and interpretable latent models of high-dimensional neural activity using pi-vae. In *Advances in Neural Information Processing Systems 33*, 2020.
6. Carlos E Vargas-Irwin, Gregory Shakhnarovich, Payman Yadollahpour, John MK Mislow, Michael J Black, and John P Donoghue. Decoding complete reach and grasp actions from local primary motor cortex populations. *Journal of neuroscience*, 30(29):9659–9669, 2010.
7. Elizaveta V Okorokova, James M. Goodman, Nicholas G. Hatsopoulos, and Sliman J. Bensmaia. Decoding hand kinematics from population responses in sensorimotor cortex during grasping. *Journal of neural engineering*, 2020.
8. Byron M. Yu, John P. Cunningham, Gopal Santhanam, Stephen I. Ryu, Krishna V. Shenoy, and Maneesh Sahani. Gaussian-process factor analysis for low-dimensional single-trial analysis of neural population activity. *Journal of neurophysiology*, 102 1:614–35, 2008.
9. MM Churchland, JP Cunningham, M. Kaufman, J. Foster, Paul Nuyujukian, Si Ryu, and K. V. Shenoy. Neural population dynamics during reaching. *Nature*, 487:51 – 56, 2012.
10. Juan Alvaro Gallego, Matthew G. Perich, Stephanie Naufel, Christian Ethier, Sara A. Solla, and Lee E. Miller. Cortical population activity within a preserved neural manifold underlies multiple motor behaviors. *Nature Communications*, 9, 2018.
11. Leland McInnes, John Healy, and James Melville. Umap: Uniform manifold approximation and projection for dimension reduction. *arXiv preprint arXiv:1802.03426*, 2018.
12. Laurens Van Der Maaten, Eric Postma, Jaap Van den Herik, et al. Dimensionality reduction: a comparative. *J Mach Learn Res*, 10 (66-71):13, 2009.
13. Geoffrey Roeder, Luke Metz, and Diederik P. Kingma. On linear identifiability of learned representations. *arXiv*, 2020. doi: 10.48550/ARXIV.2007.00810.
14. Aapo Hyvärinen, Hiroaki Sasaki, and Richard E. Turner. Nonlinear ICA using auxiliary variables and generalized contrastive learning. In *The 22nd International Conference on Artificial Intelligence and Statistics*, volume 89 of *Proceedings of Machine Learning Research*, pages 859–868. PMLR, 2019.
15. Omid G. Sani, Hamidreza Abbaspourazad, Y. Wong, Bijan Pesaran, and M. Shanechi. Modeling behaviorally relevant neural dynamics enabled by preferential subspace identification. *Nature Neuroscience*, 24:140–149, 2020.
16. David A. Klindt, Lukas Schott, Yash Sharma, Ivan Ustyuzhaninov, Wieland Brendel, Matthias Bethge, and Dylan Paiton. Towards nonlinear disentanglement in natural data with temporal sparse coding. In *International Conference on Learning Representations*, 2021.
17. Chethan Pandarinath, Daniel J. O'Shea, Jasmine Collins, Rafal Józefowicz, Sergey D. Stavisky, Jonathan C. Kao, Eric M. Trautmann, Matthew T. Kaufman, Stephen I. Ryu, Leigh R. Hochberg, Jaimie M. Henderson, Krishna V. Shenoy, L. F. Abbott, and David Sussillo. Inferring single-trial neural population dynamics using sequential auto-encoders. *Nature methods*, 15:805 – 815, 2018.
18. Luke Y. Prince, Shahab Bakhtiari, Colleen J. Gillon, and Blake A. Richards. Parallel inference of hierarchical latent dynamics in two-photon calcium imaging of neuronal populations. *bioRxiv*, 2021.
19. Michael U. Gutmann and Aapo Hyvärinen. Noise-contrastive estimation of unnormalized statistical models, with applications to natural image statistics. *The Journal of Machine Learning Research*, 13:307–361, 2012.
20. Aaron van den Oord, Yazhe Li, and Oriol Vinyals. Representation learning with contrastive predictive coding. *arXiv preprint arXiv:1807.03748*, 2018.
21. Prannay Khosla, Piotr Teterwak, Chen Wang, Aaron Sarna, Yonglong Tian, Phillip Isola, Aaron Maschinot, Ce Liu, and Dilip Krishnan. Supervised contrastive learning. *arXiv preprint arXiv:2004.11362*, 2020.
22. Ting Chen, Simon Kornblith, Mohammad Norouzi, and Geoffrey E. Hinton. A simple framework for contrastive learning of visual representations. *ArXiv*, abs/2002.05709, 2020.
23. Andres D Grosmark and György Buzsáki. Diversity in neural firing dynamics supports both rigid and learned hippocampal sequences. *Science*, 351(6280):1440–1443, 2016.
24. Hermanni Hälvä, Sylvain Le Corff, Luc Leh'ericy, Jonathan So, Yongjie Zhu, Elisabeth Gassiat, and Aapo Hyvärinen. Disentan-

gling identifiable features from noisy data with structured nonlinear ica. *ArXiv*, abs/2106.09620, 2021.
25. Roland S. Zimmermann, Yash Sharma, Steffen Schneider, Matthias Bethge, and Wieland Brendel. Contrastive learning inverts the data generating process. In *Proceedings of the 38th International Conference on Machine Learning*, volume 139 of *Proceedings of Machine Learning Research*, pages 12979–12990. PMLR, 2021.
26. John R. Huxter, Neil Burgess, and John O'Keefe. Independent rate and temporal coding in hippocampal pyramidal cells. *Nature*, 425: 828–832, 2003.
27. Edvard I. Moser, Emilio Kropff, and May-Britt Moser. Place cells, grid cells, and the brain's spatial representation system. *Annual review of neuroscience*, 31:69–89, 2008.
28. Daniel A. Dombeck, Christopher D. Harvey, Lin Tian, Loren L. Looger, and David W. Tank. Functional imaging of hippocampal place cells at cellular resolution during virtual navigation. *Nature neuroscience*, 13:1433 – 1440, 2010.
29. Raeed H Chowdhury, Joshua I Glaser, and Lee E Miller. Area 2 of primary somatosensory cortex encodes kinematics of the whole arm. *ELife*, 9:e48198, 2020.
30. Carina Curto. What can topology tell us about the neural code. *arXiv: Neurons and Cognition*, 2016.
31. Rishidev Chaudhuri, Berk Gerçek, Biraj Pandey, Adrien Peyrache, and Ila R. Fiete. The intrinsic attractor manifold and population dynamics of a canonical cognitive circuit across waking and sleep. *Nature Neuroscience*, 22:1512–1520, 2019.
32. Vin de Silva, Dmitriy Morozov, and Mikael Vejdemo-Johansson. Persistent cohomology and circular coordinates. *Discrete & Computational Geometry*, 45:737–759, 2009.
33. Richard J. Gardner, Erik Hermansen, Marius Pachitariu, Yoram Burak, Nils A. Baas, Benjamin A. Dunn, May-Britt Moser, and Edvard I. Moser. Toroidal topology of population activity in grid cells. *Nature*, 602(7895):123–128, Feb 2022. ISSN 1476-4687. doi: 10.1038/s41586-021-04268-7.
34. M. J. Prud'homme and John F. Kalaska. Proprioceptive activity in primate primary somatosensory cortex during active arm reaching movements. *Journal of neurophysiology*, 72 5:2280–301, 1994.
35. Brian M. London and Lee E. Miller. Responses of somatosensory area 2 neurons to actively and passively generated limb movements. *Journal of neurophysiology*, 109 6:1505–13, 2013.
36. Philipp Berens, Jeremy Freeman, Thomas Deneux, Nicolay Chenkov, Thomas McColgan, Artur Speiser, Jakob H. Macke, Srinivas C. Turaga, Patrick J. Mineault, Peter Rupprecht, Stephan Gerhard, Rainer W. Friedrich, Johannes Friedrich, Liam Paninski, Marius Pachitariu, Kenneth D. Harris, Ben Bolte, Timothy A. Machado, Dario L. Ringach, Jasmine Stone, Luke E. Rogerson, Nicolas J. Sofroniew, Jacob Reimer, Emmanouil Froudarakis, Thomas Euler, Miroslav Román Rosón, Lucas Theis, Andreas Savas Tolias, and Matthias Bethge. Community-based benchmarking improves spike rate inference from two-photon calcium imaging data. *PLoS Computational Biology*, 14, 2018.
37. Torkel Hafting, Marianne Fyhn, Sturla Molden, May-Britt Moser, and Edvard I. Moser. Microstructure of a spatial map in the entorhinal cortex. *Nature*, 436:801–806, 2005.
38. Wolfram Schultz, Peter Dayan, and P. Read Montague. A neural substrate of prediction and reward. *Science*, 275:1593 – 1599, 1997.
39. Jeremiah Y. Cohen, Sebastian Haesler, Linh Vong, Bradford B. Lowell, and Naoshige Uchida. Neuron-type specific signals for reward and punishment in the ventral tegmental area. *Nature*, 482:85 – 88, 2012.
40. William Menegas, Joseph F Bergan, Sachie K. Ogawa, Yoh Isogai, Kannan Umadevi Venkataraju, Pavel Osten, Naoshige Uchida, and Mitsuko Watabe-Uchida. Dopamine neurons projecting to the posterior striatum form an anatomically distinct subclass. *eLife*, 4, 2015.
41. David H. Hubel and Torsten N. Wiesel. Ferrier lecture - functional architecture of macaque monkey visual cortex. *Proceedings of the Royal Society of London. Series B. Biological Sciences*, 198:1 – 59, 1977.
42. Cristopher M. Niell, Michael P. Stryker, and Wendell M. Keck. Highly selective receptive fields in mouse visual cortex. *The Journal of Neuroscience*, 28:7520 – 7536, 2008.
43. Dario L. Ringach, Patrick J. Mineault, Elaine Tring, Nicholas D. Olivas, Pablo García-Junco-Clemente, and Joshua T. Trachtenberg. Spatial clustering of tuning in mouse primary visual cortex. *Nature Communications*, 7, 2016.
44. Saskia EJ de Vries, Jerome A Lecoq, Michael A Buice, Peter A Groblewski, Gabriel K Ocker, Michael Oliver, David Feng, Nicholas Cain, Peter Ledochowitsch, Daniel Millman, et al. A large-scale standardized physiological survey reveals functional organization of the mouse visual cortex. *Nature Neuroscience*, 23(1):138–151, 2020.
45. Joshua H. Siegle, Xiaoxuan Jia, Séverine Durand, Samuel D. Gale, Corbett Bennett, Nile Graddis, Greggory Heller, Tamina Ramirez, Hannah Choi, Jennifer A. Luviano, Peter A. Groblewski, Ruweida Ahmed, Anton Arkhipov, Amy Bernard, Yazan N. Billeh, Dillan Brown, Michael A. Buice, Nicolas Cain, Shiella Caldejon, Linzy Casal, Andrew Cho, Maggie Chvilicek, Timothy C Cox, Kael Dai, Daniel J Denman, Saskia E. J. de Vries, Roald Dietzman, Luke Esposito, Colin Farrell, David Feng, J. Galbraith, Marina Garrett, Emily C. Gelfand, Nicole Hancock, Julie A. Harris, Robert E. Howard, Brian Hu, Ross Hytnen, Ramakrishnan Iyer, Erika Jessett, Katelyn Johnson, India Kato, Justin Kiggins, Sophie Lambert, Jérôme A. Lecoq, Peter Ledochowitsch, Jung Hoon Lee, Arielle Leon, Yang Li, Elizabeth Liang, Fuhui Long, Kyla Mace, Josef Melchior, Daniel J. Millman, Tyler Mollenkopf, Chelsea Nayan, Lydia Ng, Kiet Ngo, Thuyahn Nguyen, Philip R. Nicovich, Kat North, Gabriel Koch Ocker, Douglas R. Ollerenshaw, Michael Oliver, Marius Pachitariu, Jed Perkins, Melissa Reding, David Reid, Miranda Robertson, Kara Ronellenfitch, Sam Seid, Cliff Slaughterbeck, Michelle Stoecklin, David Sullivan, Ben B. Sutton, Jackie Swapp, Carol L. Thompson, Kristen Turner, Wayne Wakeman, Jennifer D. Whitesell, Derric Williams, Ali Williford, R. D. Young, Hongkui Zeng, Sarah R. Naylor, John W. Phillips, R. Clay Reid, Stefan Mihalas, Shawn R. Olsen, and Christof Koch. Survey of spiking in the mouse visual system reveals functional hierarchy. *Nature*, 2021.
46. Mathilde Caron, Hugo Touvron, Ishan Misra, Hervé Jégou, Julien Mairal, Piotr Bojanowski, and Armand Joulin. Emerging properties in self-supervised vision transformers. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 9650–9660, 2021.
47. Kathleen Esfahany, Isabel Siergiej, Yuan Zhao, and Il Memming Park. Organization of neural population code in mouse visual system. *eNeuro*, 5, 2018.
48. Miaomiao Jin and Lindsey L. Glickfeld. Mouse higher visual areas provide both distributed and specialized contributions to visually guided behaviors. *Current Biology*, 30:4682–4692.e7, 2020.
49. Laurent Dinh, Jascha Sohl-Dickstein, and Samy Bengio. Density estimation using real nvp. *arXiv preprint arXiv:1605.08803*, 2016.
50. Felix Pei, Joel Ye, David Zoltowski, Anqi Wu, Raeed H Chowdhury, Hansem Sohn, Joseph E O'Doherty, Krishna V Shenoy, Matthew T Kaufman, Mark Churchland, et al. Neural latents benchmark'21: Evaluating latent variable models of neural population activity. *arXiv preprint arXiv:2109.04463*, 2021.
51. Daniel Deitch, Alon Rubin, and Yaniv Ziv. Representational drift in the mouse visual cortex. *Current Biology*, 31(19):4327–4339, 2021.
52. Tongzhou Wang and Phillip Isola. Understanding contrastive representation learning through alignment and uniformity on the hypersphere. In *International Conference on Machine Learning*, pages 9929–9939. PMLR, 2020.
53. Dan Hendrycks and Kevin Gimpel. Gaussian error linear units (gelus). *arXiv preprint arXiv:1606.08415*, 2016.
54. Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, Alban Desmaison, Andreas Kopf, Edward Yang, Zachary DeVito, Martin Raison, Alykhan Tejani, Sasank Chilamkurthy, Benoit Steiner, Lu Fang, Junjie Bai, and Soumith Chintala. Pytorch: An imperative style, high-performance deep learning library. In H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. Fox, and R. Garnett, editors, *Advances in Neural Information Processing Systems 32*, pages 8024–8035. Curran Associates, Inc., 2019.
55. Stéfan van der Walt, S Chris Colbert, and Gael Varoquaux. The numpy array: a structure for efficient numerical computation. *Computing in Science & Engineering*, 13(2):22–30, 2011.
56. F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay. Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research*, 12:2825–2830, 2011.
57. Pavlin G. Poličar, Martin Stražar, and Blaž Zupan. opentsne: a

modular python library for t-sne dimensionality reduction and embedding. *bioRxiv*, 2019. doi: 10.1101/731877.
58. Dmitry Kobak and George C. Linderman. Initialization is critical for preserving global data structure in both t-sne and umap. *Nature Biotechnology*, 39(2):156–157, Feb 2021. ISSN 1546-1696. doi: 10.1038/s41587-020-00809-z.
59. Christopher Tralie, Nathaniel Saul, and Rann Bar-On. Ripser.py: A lean persistent homology library for python. *The Journal of Open Source Software*, 3(29):925, Sep 2018. doi: 10.21105/joss.00925.
60. C. Tralie, T. Mease, and J.Perea. Dreimac: Dimension reduction with eilenberg-maclane coordinates. *GitHub*, 2018.
61. Dmitry Kobak, Wieland Brendel, Christos Constantinidis, Claudia E Feierstein, Adam Kepecs, Zachary F. Mainen, Xue-Lian Qi, Ranulfo Romo, Naoshige Uchida, and Christian K. Machens. Demixed principal component analysis of neural population data. *eLife*, 5, 2016.
62. Yuanjun Gao, Evan Archer, Liam Paninski, and John P. Cunningham. Linear dynamical neural population models through nonlinear embeddings. In *NIPS*, 2016.

Acknowledgments: The authors thank Matthias Bethge, Roland S. Zimmermann, Luisa Eck, Alexander Mathis, Dylan Paiton, Jakob Macke, Dmitry Kobak, Jessy Lauer, Rodrigo González, and Gary Kane for discussions and feedback on earlier versions of the manuscript or code, and the Tübingen AI Center for computing resources. Funding was provided by SNSF grant no. 310030_201057, a Novartis Foundation for Medical-Biological Research Young Investigator Grant to MWM; Google PhD Fellowship to StS; the German Academic Exchange Service (DAAD) to JHL. StS acknowledges the IMPRS-IS Tübingen and ELLIS PhD program, and JHL thanks the TUM Program in Neuroengineering. MWM is the Bertarelli Foundation Chair of Integrative Neuroscience.

Author contributions: Conceptualization: MWM, StS; Methodology: StS, JHL, MWM; Software: StS, JHL; Theory: StS; Formal analysis: StS, JHL; Investigation: StS, JHL; Data Curation: JHL, StS; Writing-Original Draft: MWM; Writing-Editing: MWM, StS, JHL.

Conflicts: StS and MWM have filed a patent pertaining to this work. The authors declare no additional conflicts of interest. The funders had no role in the conceptualization, design, data collection, analysis, decision to publish, or preparation of the manuscript.

Methods

Datasets.

Artificial Spiking Dataset. Synthetic spiking data for benchmarking in Fig. 1 was adopted from (5). The continuous 1D behavior variable $c \in [0, 2\pi)$ was sampled uniformly in the interval $[0, 2\pi)$. The true 2D latent variable $\mathbf{z} \in \mathbb{R}^2$ was then sampled from a Gaussian distribution $\mathcal{N}(\mu(c), \Sigma(c))$ with mean $\mu(c) = (c, 2\sin c)^\top$ and covariance $\Sigma(c) = \mathrm{diag}(0.6 - 0.3|\sin c|, 0.3|\sin c|)$. After sampling, the 2D latent variable $\mathbf{z}$ was mapped to spiking rates of 100 neurons by applying four randomly initialized RealNVP (49) blocks. Poisson noise was then applied (5) to map firing rates onto spike counts. The final dataset consisted of 1.5×10^4 data points, and was split into train (80%) and validation (20%) sets. We quantified consistency across the entire dataset. The additional synthetic data, used in Extended Data Fig. 1, was generated by varying the noise distribution in the above generative process. Beside Poisson noise, we used additive truncated ($[0, 1000]$) Gaussian noise with standard deviation 1 and additive uniform noise defined in $[0, 2)$ which was applied to the spiking rate. We also adapted the Poisson spiking by simulating neurons with a refractory period. For this, we scaled the spiking rates to an average rate of 110Hz. We sample inter-spike intervals from an exponential distribution with the given rate and add a refractory period of 10ms.

Rat Hippocampus Dataset. We used the dataset presented in Grosmark and Buzsáki (23). In brief, bilaterally implanted silicon-probes recorded multi-cellular electrophysiological data from the CA1 hippocampus areas from each of four male Long-Evans rats. During a given session, each rat independently ran on a 1.6 meter long linear track, where they were rewarded with water at each end of the track. The numbers of recorded putative pyramidal neurons for each rat ranged between 48 to 120. Here, we processed the data as in (5). Specifically, the spikes were binned into 25ms time windows. The position and running direction (left or right) of the rat was encoded into a 3D vector, which consisted of the continuous position value and two binary values indicating right or left direction. Recordings from each rat was parsed into trials (a round trip from one end of the track as a trial) and then split into a train, validation, and test set with a k=3 nested cross-validation scheme for the decoding task.

Macaque Dataset. We used the dataset presented in Chowdhury et al. (29). In brief, electrophysiological recordings were performed in Area 2 of somatosensory cortex (S1) in a rhesus macaque (monkey) during a center-out reaching task with a manipulandum. Specifically, the monkey performed an eight direction reaching task where on 50% of trials they actively made center-out movements to a presented target. The remaining trials were "passive" trials, where an unexpected 2N force bump was given to the manipulandum towards one of the eight target directions during a holding period. The trials were aligned as in (50), and we used the data from -100ms and 500ms from the movement onset. We used 1ms time bins and convolved the data with a Gaussian kernel with standard deviation of 40ms.

Mouse Visual Cortex Datasets. We utilized the Allen Institute 2-photon calcium imaging and Neuropixels data recorded from five mouse visual cortical areas (VISp, VISl, VISal, VISam, VISpm) and one posterior parietal cortex (PPC)-like area (VISrl) during presentation of a black-and-white movie with 30 Hz frame rate, as presented previously (44, 45, 51). For calcium imaging (2P), we used the processed dataset by de Vries et al. (44) with a sampling rate of 30 Hz, aligned to the video frames. We considered the recordings from excitatory neurons (Emx1-IRES-Cre, Slc17a7-IRES2-Cre, Cux2-CreERT2, Rorb-IRES2-Cre, Scnn1a-Tg3-Cre, Nr5a1-Cre, Rbp4-Cre_KL100, Fezf2-CreER, Tlx3-Cre_PL56) in the "Visual Coding-2P" dataset. Ten repeats of the first movie (Movie 1) were shown in all session types (A,B,C) for each mouse and we used the neurons that were recorded in all three session types, found by using the cell registration (44). The Neuropixels recordings were obtained from the "Brain Observatory 1.1" dataset (45). We used the pre-processed spike-timings and binned them to a sampling frequency of 120 Hz, aligned with the movie timestamps (i.e., exactly 4 bins are aligned with each frame). The dataset contains recordings for 10 repeats, and we used the same Movie 1 that was used for the 2P recordings. For the analysis of consistency across the visual and PPC cortical areas, we used a disjoint set of neurons for each seed to avoid higher intra-consistency due to overlapping neuron identities. We made 3 disjoint set of neurons by only considering neurons from session A (for 2P data) and non-overlapping random sampling for each seed.

CEBRA Model Framework.

Notation. We will use $\mathbf{x}, \mathbf{y}$ as general placeholder variables, and denote the multidimensional, time-varying signal as $\mathbf{s}_t$, parameterized by the time t. The multidimensional, continuous context variable $\mathbf{c}_t$ contains additional information about the experimental condition and additional recordings, similar to the discrete categorical variable k_t.

The exact composition of $\mathbf{s}$, $\mathbf{c}$ and k depends on the experimental context. CEBRA is agnostic to the exact signal types; with the default parameterizations, $\mathbf{s}_t$ and $\mathbf{c}_t$ can have up to an order of hundred or thousand dimensions. For even higher dimensional datasets (e.g. raw video, audio, ...) other optimized deep learning tools can be used for feature extraction prior to the application of CEBRA.

Applicable problem setup. We refer to $\mathbf{x} \in X$ as the *reference* sample, and to $\mathbf{y} \in Y$ as a corresponding *positive* or *negative* sample. Together, $(\mathbf{x}, \mathbf{y})$ form a positive or negative pair, based on the distribution $\mathbf{y}$ is sampled from. We denote the distribution and density function of $\mathbf{x}$ as $p(\mathbf{x})$, the conditional distribution and density of the positive sample $\mathbf{y}$ given $\mathbf{x}$ as $p(\mathbf{x}|\mathbf{y})$ and the conditional distribution and density of the negative sample $\mathbf{y}$ given $\mathbf{x}$ as $q(\mathbf{y}|\mathbf{x})$.

After sampling—and no matter whether we are considering a positive or negative pair—both samples $\mathbf{x} \in \mathbb{R}^D$ and $\mathbf{y} \in \mathbb{R}^{D'}$ are encoded by feature extractors $\mathbf{f} : X \mapsto Z$ and $\mathbf{f}' : Y \mapsto Z$. The feature extractors map both samples from signal space $X \subseteq \mathbb{R}^D, Y \subseteq \mathbb{R}^{D'}$ into a common embedding space $Z \subseteq \mathbb{R}^E$. The design and parameterization of the feature extractor is chosen by the user of the algorithm. Note that the spaces X and Y and their corresponding feature extractors can be the same (which is the case for single-session experiments in this work), but that this is not a strict requirement within the CEBRA framework (e.g., in multi-session training across animals or modalities, X and Y are selected to be signals from different mice or modalities, respectively). It is also possible to include the context variable (e.g., behavior) into X, or it is possible to set $\mathbf{x}$ to the context variable, and $\mathbf{y}$ to the signal variable.

Given two encoded samples, a similarity measure $\phi : Z \times Z \mapsto \mathbb{R}$ assigns a score to a pair of embeddings. The similarity measure needs to assign a higher score to more similar pairs of points, and have an upper bound. For this work, we consider the dot product between normalized feature vectors, $\phi(\mathbf{z}, \mathbf{z}') = \mathbf{z}^\top \mathbf{z}'/\tau$, in most analyses (latents on a hypersphere), or the negative mean squared error, $\phi(\mathbf{z}, \mathbf{z}') = -\|\mathbf{z} - \mathbf{z}'\|^2/\tau$ (latents in Euclidean space). Both metrics can be scaled by a temperature parameter τ which is either fixed, or jointly learned with the network. Other L_p norms and other similarity metrics, or even a trainable neural network (a so-called projection head commonly used in contrastive learning algorithms, cf. Hyvärinen et al. (14), Chen et al. (22)), are possible choices within the CEBRA software package. The exact choice of ϕ shapes the properties of the embedding space, and encodes assumptions about the distributions p and q.

The technique requires paired data recordings, e.g. as common in aligned time-series. The signal $\mathbf{s}_t$, continuous context $\mathbf{c}_t$ and discrete context k_t are synced in their time-point t. How the reference, positive and negative samples are constructed from these available signals is a configuration choice made by the algorithm user, and depends on the scientific question to investigate.

Optimization. Given the feature encoders $\mathbf{f}$ and $\mathbf{f}'$ for the different sample types, as well as the similarity measure ϕ, we introduce the shorthand $\psi(\mathbf{x}, \mathbf{y}) = \phi(\mathbf{f}(\mathbf{x}), \mathbf{f}'(\mathbf{y}))$. The objective function can then be compactly written as:

$$\int_{\mathbf{x} \in X} \mathrm{d}\mathbf{x} p(\mathbf{x}) \left[- \int_{\mathbf{y} \in Y} \mathrm{d}\mathbf{y} p(\mathbf{y}|\mathbf{x}) \psi(\mathbf{x}, \mathbf{y}) + \log \int_{\mathbf{y} \in Y} \mathrm{d}\mathbf{y} q(\mathbf{y}|\mathbf{x}) e^{\psi(\mathbf{x}, \mathbf{y})} \right]. \tag{1}$$

We approximate this objective (25, 52) by drawing a single positive example $\mathbf{y}_+$, and multiple *negative examples* $\mathbf{y}_i$ from the distributions outlined above, and minimize the loss function

$$\mathop{\mathbb{E}}_{\substack{\mathbf{x} \sim p(\mathbf{x}),\, \mathbf{y}_+ \sim p(\mathbf{y}|\mathbf{x}) \\ \mathbf{y}_1, \ldots, \mathbf{y}_n \sim q(\mathbf{y}|\mathbf{x})}} \left[-\psi(\mathbf{x}, \mathbf{y}_+) + \log \sum_{i=1}^{n} e^{\psi(\mathbf{x}, \mathbf{y}_i)} \right], \tag{2}$$

with a gradient-based optimization algorithm. The number of negative samples is a hyperparameter of the algorithm and larger batch sizes are generally preferable.

For sufficiently small datasets as used in this paper, both positive and negative samples are drawn from all available samples in the dataset. This is in contrast to the common practice in many contrastive learning frameworks, where a mini-batch of samples is drawn first, which are then grouped into positive and negative pairs. Allowing to access the whole dataset to form the pairs gives a better approximation of the respective distributions $p(\mathbf{y}|\mathbf{x})$ and $q(\mathbf{y}|\mathbf{x})$, and considerably improves the quality of the obtained embeddings. If the dataset is small enough to fit into the GPU memory, CEBRA can be optimized with batch gradient descent, i.e., use the whole dataset at each optimizer step.

Goodness of fit. Comparing the loss value—at both the absolute value and relative value across models at the same point in training time—can be used to determine the goodness of fit. Practically, this means one can find which hypothesis best fits one's data, in the case of using CEBRA-Behavior. Specifically, let us denote the objective in Eq. 1 as L_{asympt} and its approximation in Eq. 2 with a batch size of n as L_n. In the limit of many samples, the objective converge up to a constant, $L_{\text{asympt}} = \lim_{n \to \infty} [L_n - \log n]$ (cf. Suppl. Note 2, and Wang and Isola (52)).

The objective has also two trivial solutions: The first one is obtained for a constant $\psi(\mathbf{x}, \mathbf{y}) = \psi$, which yields a value of $L_n = \log n$. Such a solution is typically not obtained during training, as the network is initialized randomly, causing the initial embedding points to be randomly distributed in space.

If the embedding points are distributed uniformly in space, and ϕ is selected such that $\mathbb{E}[\phi(\mathbf{x}, \mathbf{y})] = 0$, we will also get a value that is *approximately* $L_n = \log n$. The value can be estimated easily by computing $\phi(\mathbf{u}, \mathbf{v})$ for randomly distributed points.

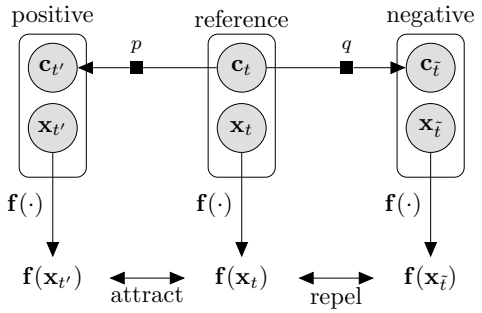
The minimizer of Eq. 1 is also clearly defined as $D_{\mathrm{KL}}(p\|q)$ and depends on the positive and negative distribution. For discovery-driven (time contrastive) learning, this value is impossible to estimate as it would require access to the underlying conditional distribution of the latents. However, for hybrid training with pre-defined positive and negative distributions, this quantity can be again numerically estimated.

Interesting values of the loss function when fitting a CEBRA model are therefore

$$-D_{\mathrm{KL}}(p\|q) \leq L_n - \log n \leq 0 \tag{3}$$

where $L_n - \log n$ is the goodness of fit (lower is better) of the CEBRA model. Note that the metric is independent of the batch size used for training.

Sampling. Selection of the sampling scheme is CEBRA's key feature to adapt embedding spaces to different datasets and recording setups. The conditional distributions $p(\mathbf{y}|\mathbf{x})$ for positive samples and $q(\mathbf{y}|\mathbf{x})$ for negative samples as well as the marginal distribution $p(\mathbf{x})$ for reference samples are specified by the user. CEBRA offers a set of pre-defined sampling techniques, but customized variants can be specified to implement additional, domain specific distributions. This form of training allows to use the context variables to shape the properties of the embedding space, as outlined in the following graphical model:



Through the choice of sampling technique, various use cases can be built into the algorithm: For instance, by forcing the positive and negative distributions to sample uniform across a factor, the model will become invariant to this factor, as including it would yield in a sub-optimal value of the objective function.

When considering different sampling mechanisms, we distinguish between *single-session* and *multi-session* datasets: A single-session dataset consists of samples $\mathbf{s}_t$, which are associated to one or more context variables $\mathbf{c}_t$ and/or k_t. These context variables allow to impose structure on the marginal and conditional distribution used for obtaining the embedding. Multi-session datasets consist of multiple single-session datasets. The dimension of context variables $\mathbf{c}_t$ and/or k_t must be shared across all sessions, while the dimension of the signal $\mathbf{s}_t$ can vary. In such a setting, CEBRA allows to learn a shared embedding space for signals from all sessions.

For single-session datasets, sampling is done in two steps: First, based on a specified "index" (the user-defined context variable $\mathbf{c}_t$ and/or k_t), locations t are sampled for reference, positive and negative samples. The algorithm differentiates between categorical (k) and continuous ($\mathbf{c}$) variables for this purpose.

In the simplest case, negative sampling (q) returns a random sample from the empirical distribution, by returning a randomly chosen index from the dataset. Optionally, with a categorical context variable $k_t \in [K]$, negative sampling can be performed to approximate a uniform distribution of samples over this context variable. If this is performed for both the negative and positive samples, the resulting embedding will become invariant with respect to the variable k_t. Sampling is performed in this case by computing the cumulative histogram of k_t, and sampling uniformly over k using the transformation theory for probability densities.

For positive pairs, different options exist based on the availability of continuous and discrete context variables. For a discrete context variable $k_t \in [K]$ with K possible values, sampling from the conditional distribution is done by filtering the whole dataset for the value k_t of the reference sample, and uniformly selecting a positive sample with the same value. For a continuous context variable $\mathbf{c}_t$, we use a set of time offsets Δ to specify the distribution. Given the time offsets, the empirical distribution $P(\mathbf{c}_{t+\tau}|\mathbf{c}_t)$ for a particular choice of $\tau \in \Delta$ can be computed from the dataset: We build up a set $D = \{t \in [T], \tau \in \Delta : \mathbf{c}_t - \mathbf{c}_{t+\tau}\}$, sample a $\mathbf{d}$ uniformly from D, and obtain the sample that is closest to the reference sample modified by this distance $\mathbf{d}$ from the dataset ($\mathbf{x} + \mathbf{d}$). It is possible to combine a continuous variable $\mathbf{c}_t$ with a categorical variable k_t for mixed sampling. On top of the continual sampling step above, it is ensured that both samples in the positive pair share the same value k_t.

It is crucial that the context samples $\mathbf{c}$ and the norm used in the algorithm match in some way; for simple context variables with predictable conditional distributions (e.g., a one or two-dimensional position of a moving animal, which can be most likely well described by a Gaussian conditional distribution based on the previous sample). An additional alternative is to use CEBRA also to *pre-process* the original context samples $\mathbf{c}$ and use the embedded context samples with the metric used for CEBRA training. This scheme is especially useful for higher dimensional behavioral data, or even complex inputs like video.

We next consider the multi-session case, where signals $\mathbf{s}_t^{(i)} \in \mathbb{R}^{n_i}$ come from N different sessions $i \in [N]$ with session-dependent dimensionality n_i. Importantly, the corresponding continuous context variables $\mathbf{c}_t^{(i)} \in \mathbb{R}^m$ share the same dimensionality m, which makes it possible to relate samples across sessions. The multi-session setup is similar to mixed session sampling (if we treat the session ID as a categorical variable $k_t^{(i)} := i$ for all time steps t in session i). The conditional distribution for both negative and positive pairs is uniformly sampled across sessions, irrespective of session length. Multi session mixed sampling or multi session discrete sampling can be implemented analogously.

Besides the outlined sampling scheme, CEBRA is flexible to incorporate more specialized sampling schemes. For instance, mixed single session sampling could be extended to additionally incorporate a dimension the algorithm should become invariant to. This would add an additional step of uniform sampling with regard to to this desired discrete variable (e.g., via ancestral sampling).

Choice of reference and positive and negative samples. Depending on the exact application, the contrastive learning step can be performed by explicitly including or excluding the context variable: The reference sample $\mathbf{x}$ can contain information from the signal $\mathbf{s}_t$, but also from the experimental conditions, behavioral recordings, or other context variables. The positive and negative samples $\mathbf{y}$ are set to the signal variable $\mathbf{s}_t$.

Theoretical guarantees for linear identifiability of CEBRA models. Identifiability describes the property of an algorithm to give a consistent estimate for the model parameters given that the data distributions match. We here apply the relaxed notion of *linear identifiability* that was previously discussed and used by Hyvärinen et al. (14) and Roeder et al. (13): After training two encoder models $\mathbf{f}$ and $\mathbf{f}'$, the models are linear identifiable if $\mathbf{f}(\mathbf{x}) = \mathbf{L}\mathbf{f}(\mathbf{x})$ where $\mathbf{L}$ is a linear projection.

When applying CEBRA, three cases are of potential interest. First, when applying discovery-driven CEBRA, will two models estimated on comparable experimental data agree in their inferred representation? Second, under which assumptions about the data will we be able to discover the *true* latent distribution? Third, in the hypothesis-driven or hybrid application of CEBRA, is the algorithm guaranteed to give a meaningful (non-standard) latent space when we can find signal within the data?

For the first case, we note that the CEBRA objective with a cosine similarity metric follows the canonical discriminative form for which Roeder et al. (13) showed linear identifiability: For sufficiently diverse datasets, two CEBRA models trained to convergence on the same dataset will be consistent up to linear transformations. Note that consistency of CEBRA is independent of the exact data distribution: The diversity condition merely requires that for any set of samples $\{\mathbf{y}_1, \ldots, \mathbf{y}_d\}$ from the negative distribution $q(\cdot|\mathbf{x})$, the matrices $[\cdots \mathbf{f}'(\mathbf{y}_i) - \mathbf{f}'(\mathbf{y}_i) \cdots]_{i=1}^{d}$ and the matrix $[\cdots \mathbf{f}(\mathbf{x}_i) \cdots]_{i=1}^{d+1}$ are invertible (i.e., the embeddings are sufficiently diverse). Alternatively, we can derive linear identifiability from assumptions about the data distribution: If the ground truth latents are sufficiently diverse (i.e., vary in all directions under the distributions p and q), and the model is sufficiently parameterized to fit the data, we will also obtain consistency up to a linear transformation. See Suppl. Note 2 for a full formal discussion and proofs.

For the second case, additional assumptions are required regarding the exact form of the data generating distribution. Within the scope of this work, we consider ground truth latents distributed on the hypersphere or Euclidean space. The metric then needs to match assumptions about the variation of the ground truth latents over time. In discovery-driven CEBRA, using the dot product as the similarity measure then encodes the assumption that latents vary according to a von-Mises-Fisher distribution, while the mean squared error encodes an assumption that latents vary according to a Normal distribution. More broadly, if we assume that the latents have a uniform marginal distribution (which can be ensured by designing un-biased experiments), the similarity measure should be chosen as the log-likelihood of the conditional distribution over time. In this case, CEBRA identifies the data generating distribution up to affine transforms (in the most general case).

This result also explains the empirically high performance of CEBRA for decoding applications: If trained for decoding (using the variable to decode for informing the conditional distribution), it is trivial to select matching conditional distributions, as both quantities are directly selected by the user. CEBRA then "identifies" the context variable up to a linear transformation.

For the third case, we are interested in the hypothesis testing capabilities. We can show that if a mapping exists between the context variable and the signal space, CEBRA will recover this relationship and yield a meaningful embedding, which is also decodable. However, if such a mapping does not exist, we can show that CEBRA will instead learn a default embedding which is the uniform distribution of points on the hypersphere.

CEBRA models.

We ran all experiments using our PyTorch implementation of CEBRA. We chose $X = Y$ to be the neural signal with varying amounts of recorded neurons and channels based on the dataset. We used three types of encoder models based on the required receptive field; a receptive field of one sample was used on the synthetic dataset experiments (Fig. 1b), a receptive field of 10 samples in all other experiments (rat, monkey, mouse) except for the Neuropixels dataset, where a receptive field of 40 samples is used due to the 4 times higher sampling rate of the dataset.

All feature encoders are parameterized by the number of neurons (input dimension), a hidden dimension to control the model size and capacity, as well as their output (embedding) dimension. For the model with the receptive field of one, a four layer MLP was used. The first and second layers map their respective inputs to the hidden dimension, while the third layer introduces a bottleneck and maps to half the hidden dimension. The final layer maps to the requested output dimension. For the model with receptive field of 10, a convolutional network with five time convolutional layers was used. The first layer had kernel size 2, the next three layers had kernel size 3 and used skip connections. The final layer had kernel size 3 and mapped the hidden dimensions to the output dimension. For the model with receptive field 40, we first preprocessed the signal by concatenating a $2\times$ downsampled version of the signal with a learnable downsample operation implemented as a convolutional layer with kernel size 4 and stride 2, directly followed (without activation function in between) by another convolutional layer with kernel size 3 and stride 2. After these first layers, the signal is subsampled by a factor of 4. Afterwards, similar to the receptive field 10 model, we apply three layers with kernel size 3 and skip connections, and a final layer with kernel size 3. In all models, Gaussian error linear unit activation functions (GELU; 53) were applied after each layer except the last. The feature vector was normalized after the last layer, unless a mean squared error (MSE) based similarity metric was used (as in Extended Data Fig S8).

Our implementation of the InfoNCE criterion received a mini-batch (or the full dataset) of size $n \times d$ for each of the reference, positive, and negative samples. n dot-product similarities are computed between reference and positive samples, $n \times n$ dot-product similarities are computed between reference and negative samples. The similarities were scaled with the inverse of the temperature parameter τ.

```
from torch import einsum, logsumexp, no_grad

def info_nce(ref, pos, neg, τ = 1.0):
  pos_dist = einsum("nd,nd->n",  ref, pos) / τ
  neg_dist = einsum("nd,md->nm", ref, neg) / τ
  with no_grad():
    c, _ = neg_dist.max(dim=1)
  pos_dist = pos_dist - c.detach()
  neg_dist = neg_dist - c.detach()
  pos_loss = -pos_dist.mean()
  neg_loss = logsumexp(neg_dist, dim=1).mean()
  return pos_loss + neg_loss
```

Alternatively, a learnable temperature can be used. For a numerically stable implementation, we store the log inverse temperature $\alpha = -\log \tau$ as a parameter of the loss function. At each step, we scale the distances in the loss function with $\min(\exp \alpha, 1/\tau_{\min})$. The additional parameter $\tau_{\min}$ is a lower bound on the temperature. The inverse temperature used for scaling the distances in the loss will hence lie in $(0, 1/\tau_{\min}]$.

CEBRA Model parameters used. In the main figures we used the default parameters (see https://cebra.ai/docs/api.html) for fitting CEBRA unless otherwise stated in the text (such as dimension, which varied and is noted in figures), or below.

Synthetic data: `model_architecture='offset1-model-mse'`, `conditional='delta'`, `delta=0.1`, `distance='euclidean'`, `batch_size=512`, `learning_rate=1e-4`.

Rat hippocampus: `model_architecture='offset10-model'`, `time_offsets=10`, `batch_size=512`.

Primate S1: `model_architecture='offset10-model'`, `time_offsets=10`, `batch_size=512`.

Allen datasets (2P): `model_architecture='offset10-model'`, `time_offsets=10`, `batch_size=512`.

Allen datasets (NP): `model_architecture='offset40-model-4x-subsample'`, `time_offsets=10`, `batch_size=512`.

CEBRA API and example usage. The Python implementation of CEBRA is written in PyTorch (54) and NumPy (55) and provides an API which is fully compatible with scikit-learn (56), a commonly used package for machine learning. This allows to use scikit-learn tools for hyperparameter selection and downstream processing of the embeddings, e.g., decoding. CEBRA can be used as a drop-in replacement in existing data pipelines for algorithms like tSNE, UMAP, PCA or FastICA. Both CPU and GPU implementations are available.

Using the previously introduced notations, suppose we have a dataset containing signals $\mathbf{s}_t$, continuous context variables $\mathbf{c}_t$ and discrete context variables k_t for all time steps t,

```
import numpy as np
N = 500
s = np.zeros((N, 55), dtype=float)
k = np.zeros((N,), dtype=int)
c = np.zeros((N, 10), dtype=float)
```

along with a second session of data,

```
s2 = np.zeros((N, 75), dtype=float)
c2 = np.zeros((N, 10), dtype=float)
assert c2.shape[1] == c.shape[1]
```

and note that the number of samples as well as the dimension in $\mathbf{s}'$ does not need to match $\mathbf{s}$. Session alignment leverages the fact that the second dimension of $\mathbf{c}$ and $\mathbf{c}'$ match. With this dataset in place, different variants of CEBRA can be applied as follows:

```
import cebra
model = cebra.CEBRA(
    output_dimension=8,
    num_hidden_units=32,
    batch_size=1024,
    learning_rate=3e-4,
    max_iterations=1000
)
```

The training mode to use is determined automatically based on what combination of data is passed to the algorithm:

```
# time contrastive learning
model.fit(s)
# discrete behavior contrastive learning
model.fit(s, k)
# continuous behavior contrastive learning
model.fit(s, c)
# mixed behavior contrastive learning
model.fit(s, c, k)
# multi-session training
model.fit([s, s2], [c, c2])
# adapt to new session
model.fit(s, c)
model.fit(s2, c2, adapt = True)
```

Since CEBRA is a parametric method training a neural network internally, it is possible to embed new data points after fitting the model:

```
s_test = np.zeros((N, 55), dtype=float)
# obtain and plot embedding
z = model.transform(s_test)
plt.scatter(z[:, 0], z[:, 1])
plt.show()
```

Consistency of embeddings across runs, subjects, sessions, recording modalities, and areas.

To measure the consistency of the embeddings, we used the R^2 score of the linear regression (including an intercept) between the embeddings from different subjects (or sessions). Secondly, pi-VAE, which we benchmarked and improved (Extended Data Fig. S1), demonstrated a theoretical guarantee that it can reconstruct the true latent space up to an affine transformation. To measure across runs, we measured the R^2 score of the linear regression between embeddings across 10 runs of the algorithms, yielding 90 comparisons. The runs were done with the same hyperparameters, model, and training setup.

For the rat hippocampus data, the number of neurons recorded were different across subjects. The behavior setting was the same: the rats moved in a 1.6 meter long track, and for analysis the behavior data was binned into 100 bins with equal size for each direction (leftwards, rightwards). We computed averaged feature vectors for each bin by averaging all normalized CEBRA embeddings for a given bin, and re-normalized the average to lie on the hypersphere. If a bin does not contain any sample, it was filled by samples from the two adjacent bins. CEBRA was trained with latent dimension 3 (the minimum) such that it is constrained to lie only on a 2-sphere (making this "3D" space equivalent to 2D Euclidean space). All other methods were trained with 2 latent dimensions in Euclidean space. Note that $n+1$ dimensions of CEBRA is equivalent to n dimensions of other methods that we compared, since the feature space of CEBRA is normalized (i.e., the feature vectors are normalized to have unit length).

For Allen visual data where the number of behavioral data points are the same across different sessions (i.e., fixed length of video stimuli), we directly computed the R^2 score of linear regression between embeddings from different sessions and the modalities. We surveyed 3, 4, 8, 32, 64, 128 latent dimensions with CEBRA.

To compare the consistency of embeddings between or within the areas we considered, we computed intra-area and inter-area consistency within the same recording modality (2P or NP). Within the same modality, we sampled 400 neurons from each area. We trained one CEBRA model per area, and computed the linear consistency between all pairs of embeddings. For the intra-area comparison, we sampled an additional 400 disjoint neurons. For each area, we trained two CEBRA models on these two sets of neurons, and computed their linear consistency. We repeated this process three times. For comparisons across modalities (2P and NP), we sampled 400 neurons from each modality (which are disjoint, as above, because one set was sampled from 2P recordings and the other set from the NP recordings). We trained a multi-session CEBRA model with one encoder for 2P, and one encoder for NP in the same embedding space. For an intra-area comparison, we computed the linear consistency between the the NP and 2P decoder from the same area. For an inter-area comparison, we computed the linear consistency between the NP encoder from one area and the 2P encoder from another area and again considered all combinations of areas. We repeated this process three times.

For the comparison of single- and multi-session training (Extended Data Fig. S7), we computed embeddings using encoder models with 8, 16, ..., 128 hidden units for varying the model size, and benchmark 8, 16, ..., 128 latent dimensions. Hyperparameters, except for number of optimization steps, were selected according to validation set decoding R^2 (rat) or accuracy (Allen). Consistency is reported at the point in training where the position decoding error is less than 7 cm for the first rat in the hippocampus dataset, and a decoding accuracy of 60% on the Allen dataset. For single-session training,

four embeddings were trained independently on each of the individual animals, while for multi-session the embeddings were trained jointly on all sessions. For multi-session training, the same number of samples was drawn from each session to learn an embedding invariant to the session ID. The consistency vs. decoding error trade-off (Extended Data Fig. S7c) was reported as the average consistency across all 12 comparisons (Extended Data Fig. S7b) vs. the average decoding performance across all rats and data splits.

Model Comparisons.

pi-VAE parameter selection, and modifications to pi-VAE. The original implementation of pi-VAE used a single time bin spiking rate as a input. Thus, we modified their code to allow for larger time bin inputs and found that time window input with receptive field of 10 time bins (250 ms) gave a higher consistency across subjects and better preserved the qualitative structure of the embedding (thereby outperforming the results presented in (5); see Extended Data Fig. S1). To do this, we used the same encoder neural network architecture as we used for CEBRA, and modified the decoder to a 2D output (we call our modified version conv-pi-VAE). Note, we used this modified pi-VAE for all the experiments except for the synthetic setting, where there is no time dimension, thus the original implementation is sufficient.

The original implementation reported a median absolute error of 12 cm on rat 1 (the animal they considered most in the work), and our implementation of time windowed input with 10 bins resulted in a median absolute error of 11 cm (Fig. 2). For hyperparameters, we tested different epochs between 600 (the published value used) and 1000, and learning rate between 1.0×10^{-6} and 5.0×10^{-4} via a grid search. We fixed the hyperparameters to be those that gave the highest consistency across subjects, which were training epochs of 1000 and learning rate 2.5×10^{-4}. All other hyperparameters were kept as in the original implementation (5). Note, that the original paper demonstrated that pi-VAE is fairly robust across different hyperparameters. For decoding (Fig. 2) we considered both a simple kNN decoder (that we use for CEBRA) and the computationally more expensive Monte Carlo sampling method originally proposed for pi-VAE (5). Our implementation of conv-pi-VAE can be found at: https://github.com/AdaptiveMotorControlLab/CEBRA.

UMAP parameter selection. For UMAP (11), following the parameter guide (umap-learn.readthedocs.io/), we focused on tuning the number of neighbors ($n\_neighbors$) and minimum distance (min_dist). The $n\_components$ parameter was fixed to 2 and we used a cosine metric to make a fair comparison with CEBRA, which also used the cosine distance metric for learning. We performed a grid search with 100 total hyperparameter values in the range of [2, 200] for $n\_neighbors$ and range of [0.0001, 0.99] for min_dist. The highest consistency across runs in the rat hippocampus dataset was achieved with min_dist of 0.0001 and $n\_neighbors$ of 24. For the other datasets in Extended Data Fig. S3, we used the default value of $n\_neighbors$ as 15 and min_dist as 0.1.

tSNE parameter selection. For tSNE (12), we used the implementation in openTSNE (57). We performed a sweep on $perplexity$ in the range of [5, 50] and $early\_exaggeration$ in the range [12, 32] following the parameter guide, while fixing $n\_components$ as 2 and used a cosine metric, to fairly compare to UMAP and CEBRA. We use PCA initialization to improve the run consistency of tSNE (58). The highest consistency across runs in the rat hippocampus dataset was achieved with $perplexity$ of 10 and $early\_exaggeration$ of 16.44. For the other datasets in Extended Data Fig. S3, we used the default value of $perplexity$ of 30 and $early\_exaggeration$ of 12.

Decoding Analysis.

We primarily used a simple k-Nearest Neighbors (kNN) algorithm, which is a non-parametric supervised learning method, as a decoding method for CEBRA. We used the implementation in scikit-learn (56). We used a kNN regressor for continuous value regression and a kNN classifier for discrete label classification, using uniform weights on distances of k-nearest neighbors. For the embeddings obtained with cosine metrics, we used cosine distance metrics for kNN and Euclidean distance metrics for the embeddings obtained in Euclidean space.

For the rat hippocampus data, a kNN regressor, as implemented in scikit-learn (56), was used to decode the position, and a kNN classifier to decode the direction. The number of neighbors was searched over the range $\{1, 4, 9, 16, 25\}$ and we used the cosine distance metric. We used the R^2 score of predicted position and direction vector on the validation set as a metric to choose the best $n\_neighbors$ parameter. We report the median absolute error (MAE) for the positional decoding on the test set. For pi-VAE, we additionally evaluate decoding quality using the originally proposed decoding method based on Monte Carlo sampling, using the settings from the original paper (5). Note, UMAP, tSNE and CEBRA-Time were trained using the full dataset without label information when learning the embedding, and we used the above split only for training and cross-validation of the decoder.

For the direction decoding within the monkey dataset, we used a Ridge classifier (56) as a baseline. The regularization hyperparameter was searched over $[10^{-6}, 10^{2}]$. For CEBRA, we used a kNN classifier for decoding direction with k searched over the range [1, 2500]. For conv-pi-VAE, we searched for the best learning rate over $[1.0 \times 10^{-5}, 1.0 \times 10^{-3}]$. For position decoding, we used Lasso (56) as a baseline. The regularization hyperparameter was searched over $[10^{-6}, 10^{2}]$. For conv-pi-VAE, we used 600 epochs and searched for the best learning rates over $[5 \times 10^{-4}, 2.5 \times 10^{-4}, 0.125 \times 10^{-4}, 5 \times 10^{-5}]$, via a grid of (x,y) space in 1 cm bin for each axis as the sampling process for decoding. For CEBRA, we used the kNN regression, and the number of neighbors k was again searched over [1, 2500].

For the Allen Institute datasets, we performed decoding (frame number or scene classification) for each frame from Movie 1. Here, we used a kNN classifier (56) with a population vector kNN as a baseline, similar to the decoding of orientation grating as performed in (44). For CEBRA, we used the same kNN classifier method on the CEBRA features. In both cases, the number of neighbors k was searched over a range of [1, 100] in an exponential fashion. We used the neural data recorded during the first 8 repeats as the train set, and the 9^{th} repeat for validation to choose the hyperparameter, and the last repeat as the test set to report the decoding accuracy. We also used a Gaussian Naive Bayes decoder (56) to test linear decoding from the CEBRA model and neural population vector. Here, we assumed uniform priors over frame number and searched over a range of $[10^{-10}, 10^{3}]$ in an exponential manner for $smoothingvar$ hyperparameter.

For layer specific decoding we used data from excitatory neurons in area VISp: layer 2/3 [Emx1-IRES-Cre, Slc17a7-IRES2-Cre]; layer 4 [Cux2-CreERT2, Rorb-IRES2-Cre, Scnn1a-Tg3-Cre]; layer 5/6 [Nr5a1-Cre, Rbp4-Cre_KL100, Fezf2-CreER, Tlx3-Cre_PL56, Ntrsr1-cre].

Topological Analysis.

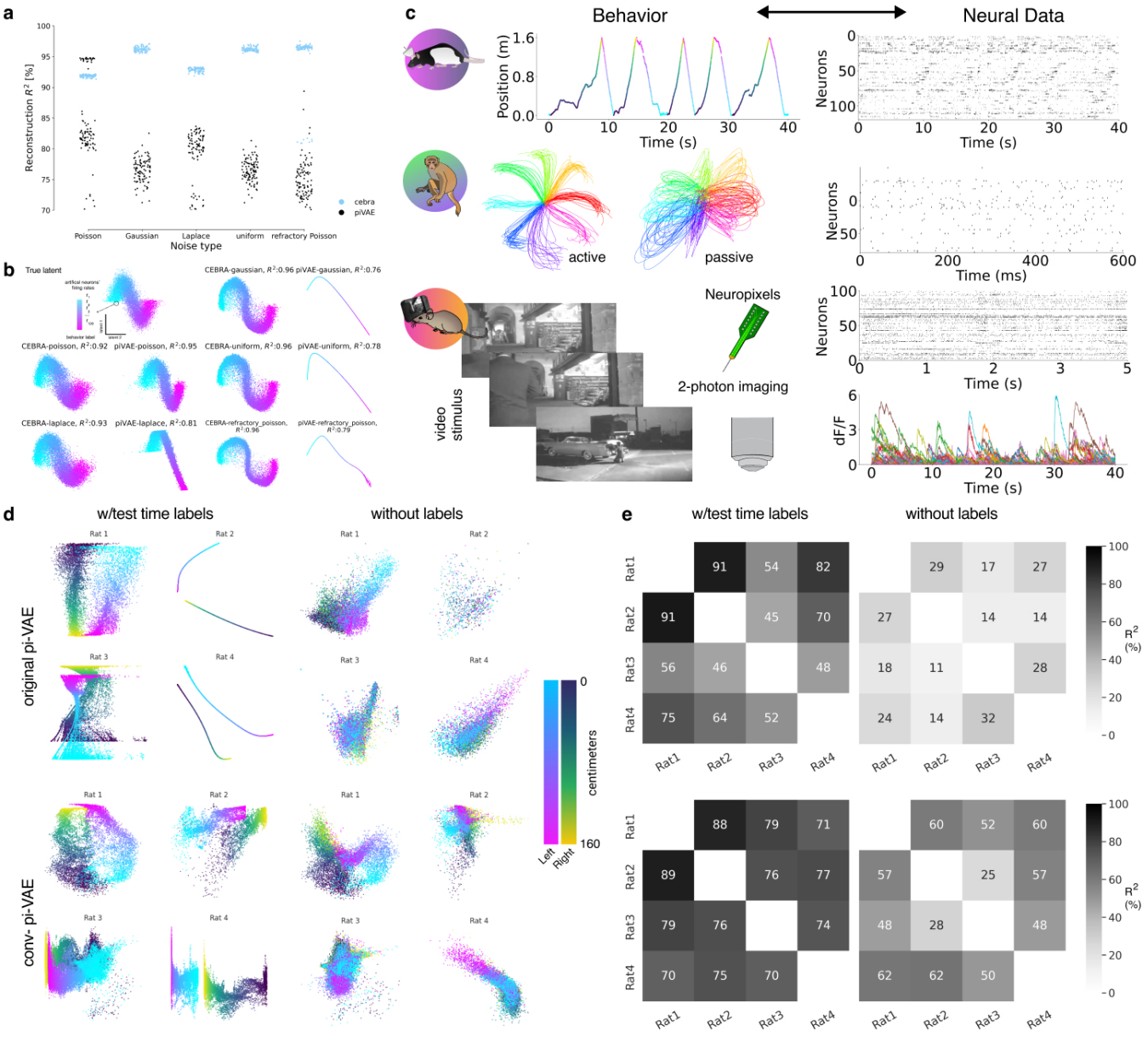
For the persistent co-homology analysis, we utilized ripser.py (59). For the hippocampus dataset we used 1,000 randomly sampled points from CEBRA-Behavior trained with temperature 1, time offset 10 and mini-batch size 512 for 10k training steps on the full dataset, and then analyzed up to the 2D co-homology. Maximum distance considered for filtration was set to infinity. To decide the number of co-cycles in each co-homology dimension with a significant lifespan, we trained 500 CEBRA embeddings with shuffled labels, similar to the approach in (33). We took the maximum lifespan of each dimension across these 500 runs as a threshold to determine robust Betti numbers. We surveyed the Betti numbers of CEBRA embeddings across 3, 8, 16, 32, and 64 latent dimensions.

Next, we used DREiMac (60) to obtain topology-preserving circular coordinates (radial angle) of the first co-cycle (H_1) from the persistent co-homology analysis. Similar to above, we used 1,000 randomly sampled points from the CEBRA-Behavior models of embedding dimensions 3, 8, 16, 32 and 64.

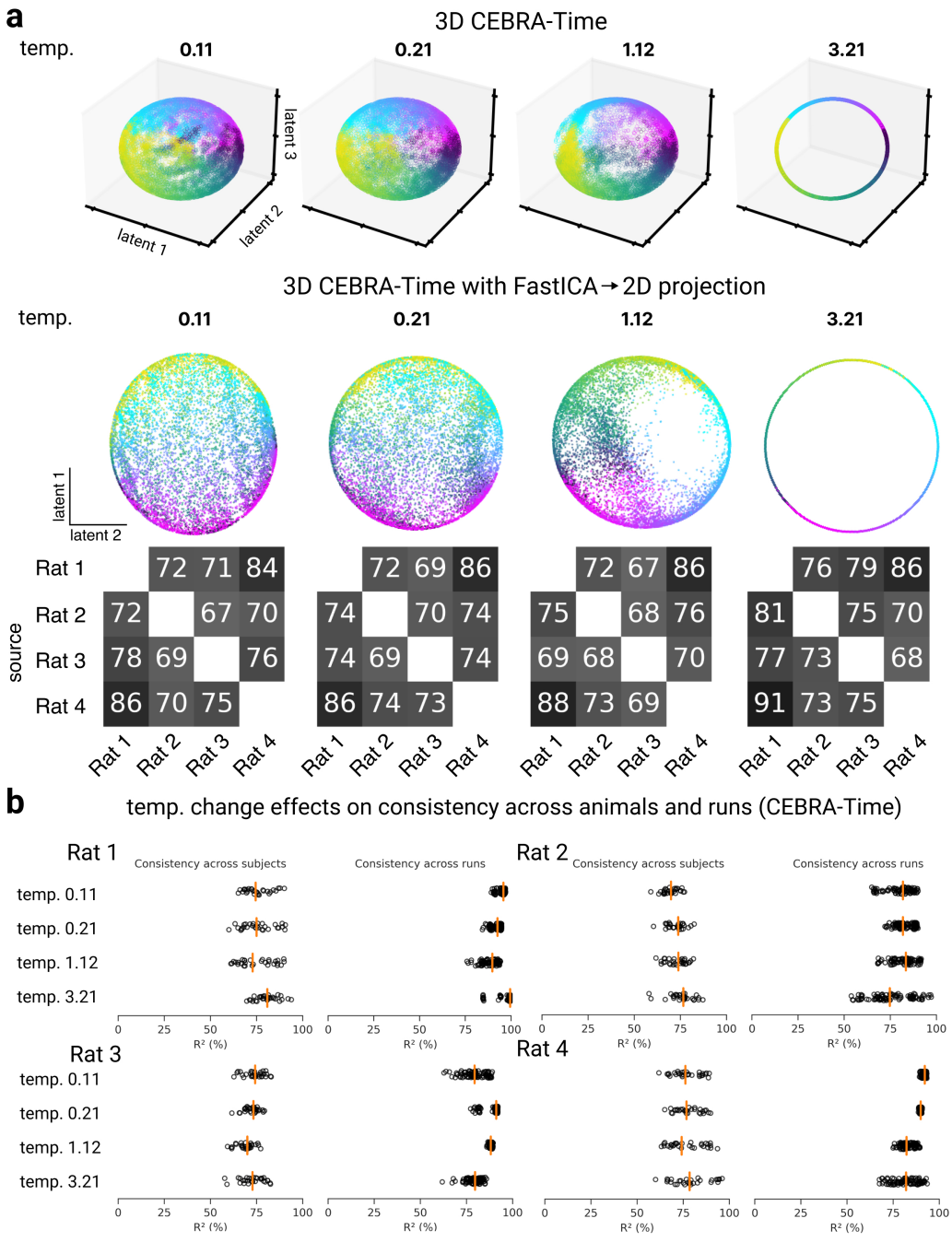
Behavior Embeddings for Video Datasets.

High dimensional inputs, such as videos, need further preprocessing for effective use with CEBRA. Firstly, we used the recently presented DINO model (46) to embed video frames into a 768-dimensional feature space. Specifically, we used the pretrained ViT/8 vision transformer model, which was trained by a self-supervised learning objective on the ImageNet database. This model is particularly well-suited for video analysis, and among the state-of-the-art models for embedding natural images into a space appropriate for k-nearest neighbour search (46), a desired property to make the dataset compatible with CEBRA. We obtained a normalized feature vector for each video frame, which was then used as the continuous behavior variable for all further CEBRA experiments.

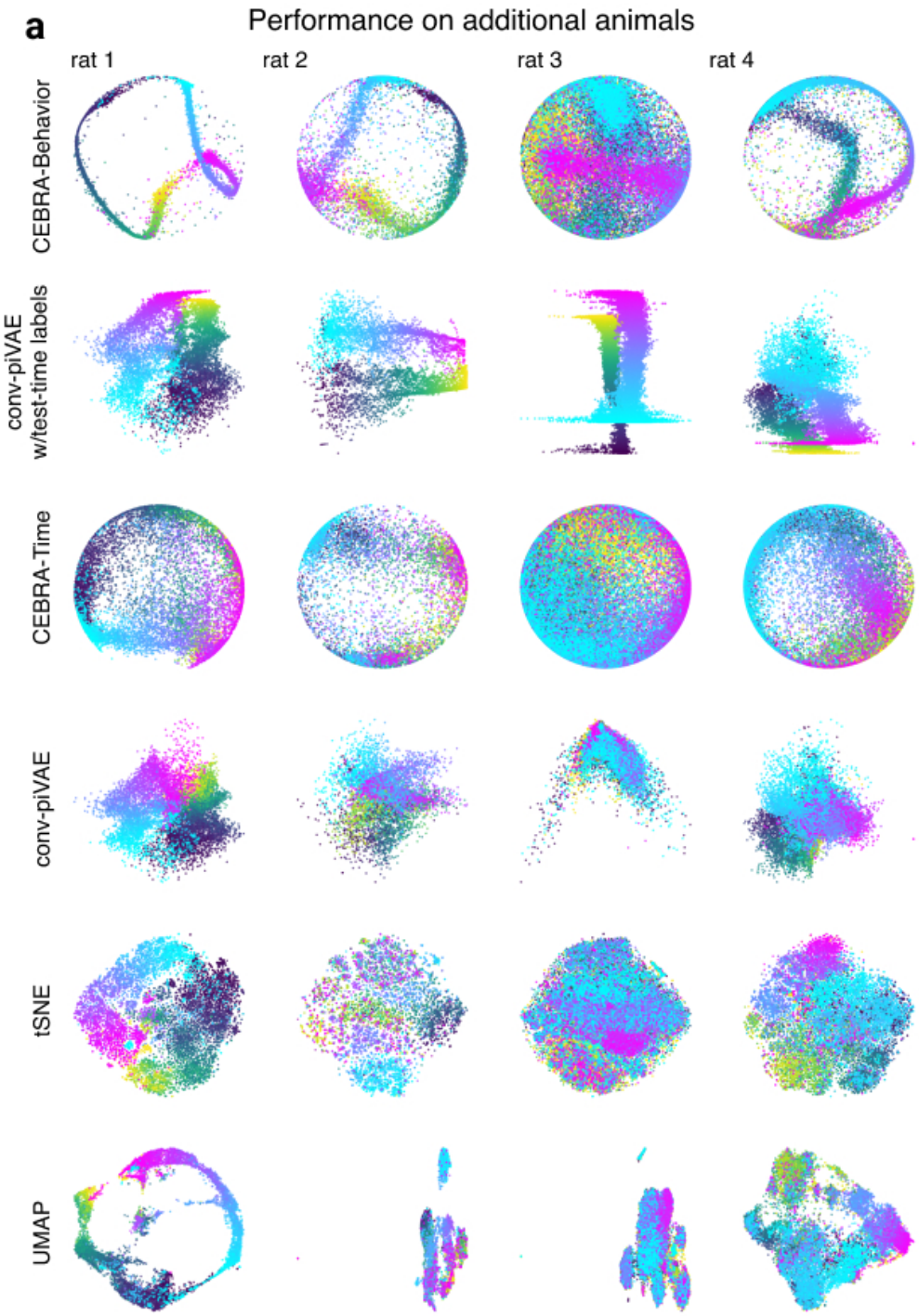
For scene labels, 3 individuals labeled each video frame using 8 candidate descriptive labels allowing multi-label classes. We took the majority vote of the 3 individuals to decide the label of each frame. In case of multi-labels, we considered this as a new class label. The above procedure resulted in 10 classes of frame annotation.



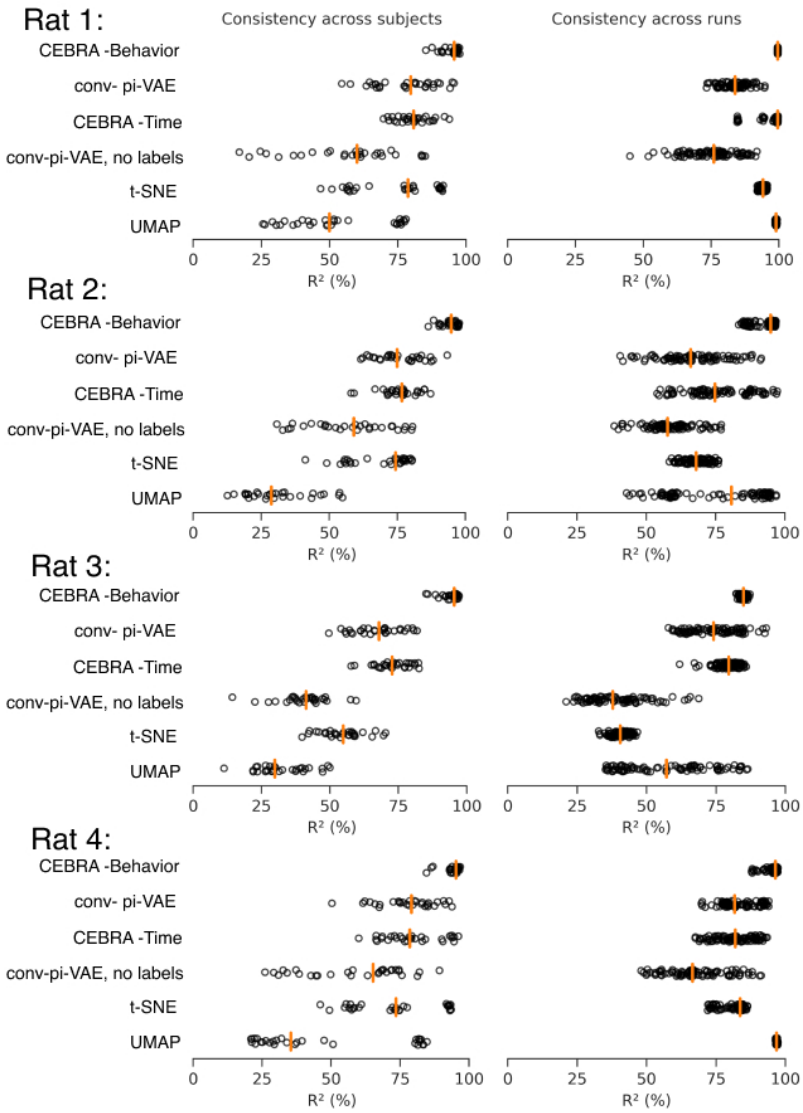
Extended Data Fig. S1. Overview of datasets, synthetic data, & original pi-VAE implementation vs. modified conv-pi-VAE. ab): We generated synthetic datasets similar to Fig. 1b with additional variations in the noise distributions in the generative process. We benchmarked the reconstruction score of the true latent using CEBRA and pi-VAE (100 seeds) on the generated synthetic datasets. CEBRA showed higher and less variable reconstruction scores than pi-VAE in all noise types. **(b)** Example visualization of the reconstructed latents from CEBRA and pi-VAE on different synthetic dataset types. **(c)**: we benchmarked and demonstrate the abilities of CEBRA on four datasets. Rat-based electrophysiology data from Grosmark and Buzsáki (23), where the animal transversed a 1.6m linear track "leftwards" or "rightwards". Two mouse-based datasets: one 2-photon calcium imaging passively viewing dataset from de Vries et al. (44), and one with the same stimulus but recorded with Neuropixels (45). A monkey-based electrophysiology dataset of center out reaching from Chowdhury et al. (29), and processed to trial data as in Pei et al. (50). **(d)**: Conv-pi-VAE showed improved performance, both with labels (Wilcoxon signed-rank test, p=0.0341) and without labels Wilcoxon signed-rank test, p=0.0005). Example runs/embeddings the consistency across rats, with **(e)**: consistency across rats, from target to source, as computed in Fig. 1.



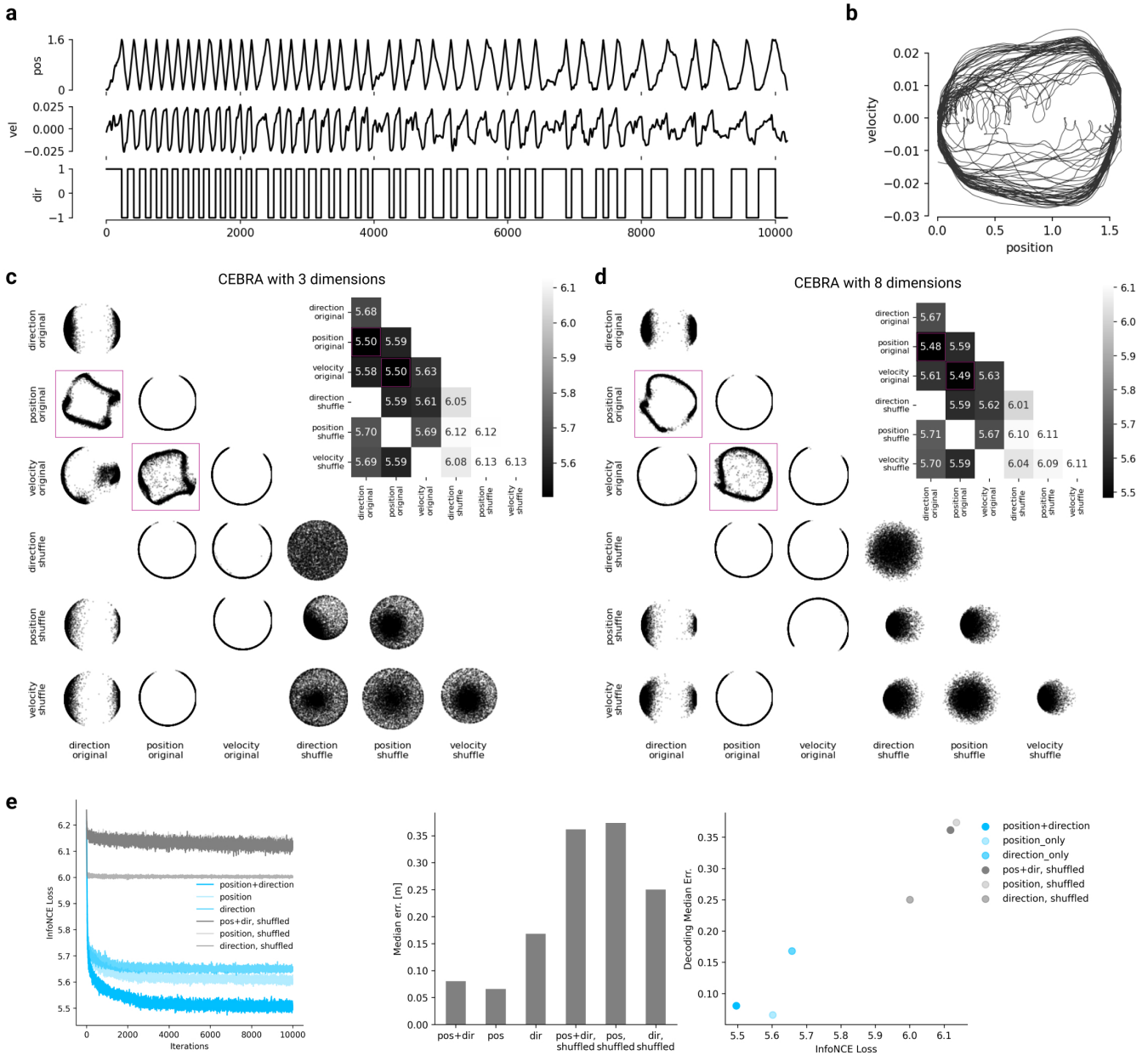
Extended Data Fig. S2. Hyperparameter changes on visualization and consistency. (a): Temperature has the largest effect on visualization (vs. consistency) of the embedding as shown by a range from 0.1 to 3.21 (highest consistency for Rat 1), as can be appreciated in 3D (top) and post FastICA into a 2D embedding (middle). Bottom row shows the corresponding change on mean consistency, and in **b,** the variance can be noted. Orange line denotes the median and black dots are individual runs (subject consistency: 10 runs with 3 comparisons per rat; run consistency: 10 runs, each compared to 9 remaining runs).



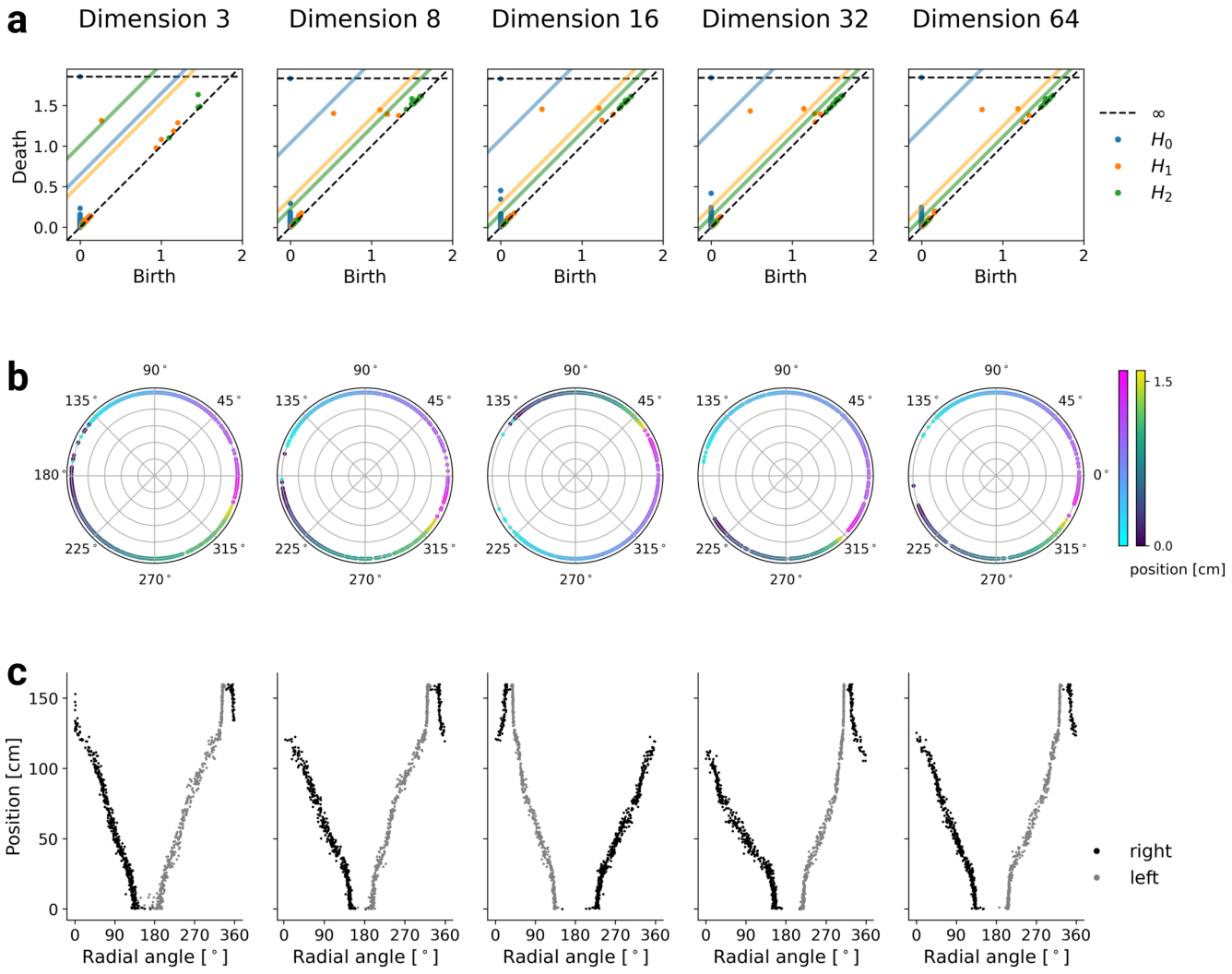
Extended Data Fig. S3. CEBRA produced consistent, highly decodable embeddings (a): Additional rat data shown for all algorithms we benchmarked (see Methods). For CEBRA-Behavior, we used temperature 1, time offset 10, batch size 512 and 10k training steps. For CEBRA-Time, we used temperature 2.25, time offset 10, batch size 512 and 4k training steps. For UMAP, we used the cosine metric and min_dist of 0.99 and $n\_neighbors$ of 31. For tSNE we used cosine metric and $perplexity$ of 29. For conv-pi-VAE, we trained 1000 epochs with learning rate 2.5×10^{-4}. CEBRA was trained with output latent 3D (the minimum) and all other methods were trained with a 2D latent.



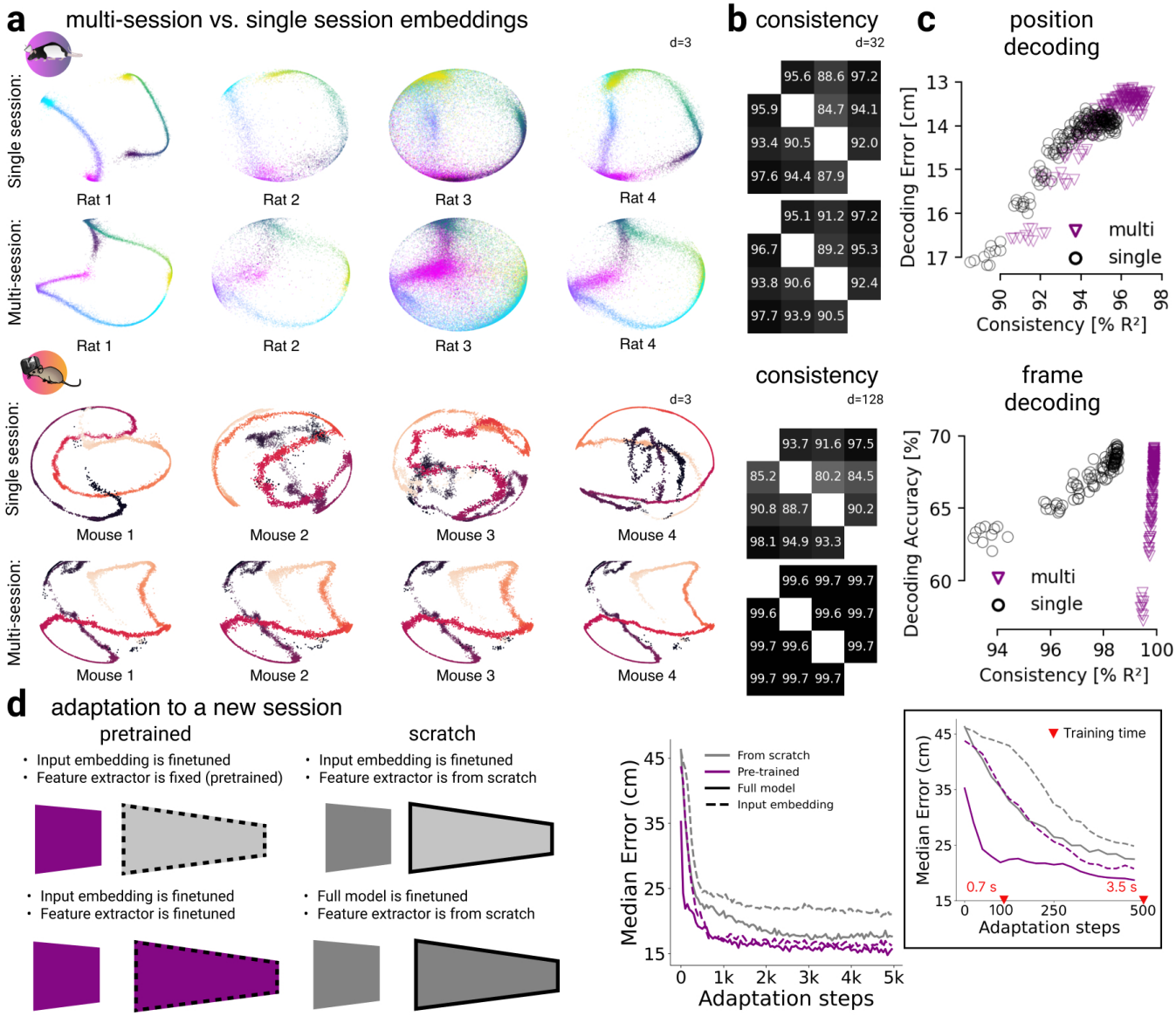
Extended Data Fig. S4. Additional metrics used for benchmarking consistency (a): Comparisons of all algorithms along different metrics for Rats 1, 2, 3, 4. The orange line is median across n=10 runs, black circles denote individual runs. Each run is the average over three non-overlapping test splits.



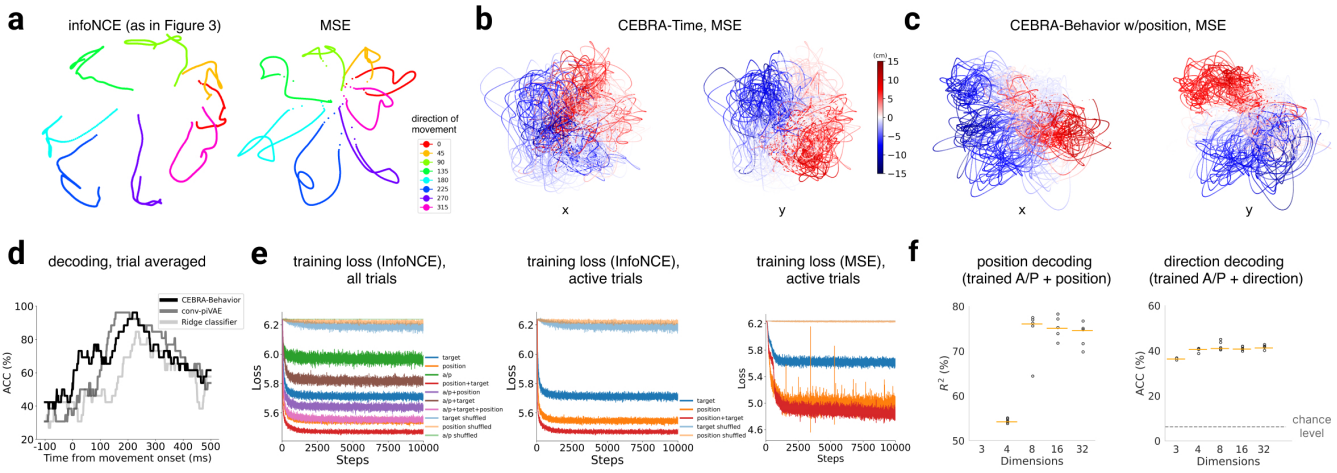
Extended Data Fig. S5. Hypothesis testing with CEBRA (a): Example data from a hippocampus recording session (Rat 1). We tested possible relationships between three experimental variables (rat location, velocity, movement direction) and the neural recordings (120 neurons, not shown). **(b)**: Relationship between velocity and position. **(c)**: We trained CEBRA with three-dimensional outputs on every single experimental variable (main diagonal) and every combination of two variables. All variables are treated as "continuous" in this experiment. We compared original to shuffled variables (shuffling is done by permuting all samples over the time dimension) as a control. We projected the original three dimensional space onto the first principal components. We show the minimum value of the InfoNCE loss on the trained embedding for all combinations in the confusion matrix (lower number is better). Either velocity or direction, paired with position information is needed for maximum structure in the embedding (highlighted, colored), yielding lowest InfoNCE error. **(d)**: Using an eight-dimensional CEBRA embedding did not qualitatively alter the results. We again report the first two principal components as well as InfoNCE training error upon convergence, and find non-trivial embeddings with lowest training error for combinations of direction/velocity and position. **(e)**: The InfoNCE metric can serve as the goodness of fit metric, both for hypothesis testing and identifying decodable embeddings. We trained CEBRA in discovery-driven mode with 32 latent dimensions (empirically the best setup for decoding). We compared the InfoNCE loss (left, middle) between various hypotheses. Low InfoNCE was correlated with low decoding error (right).



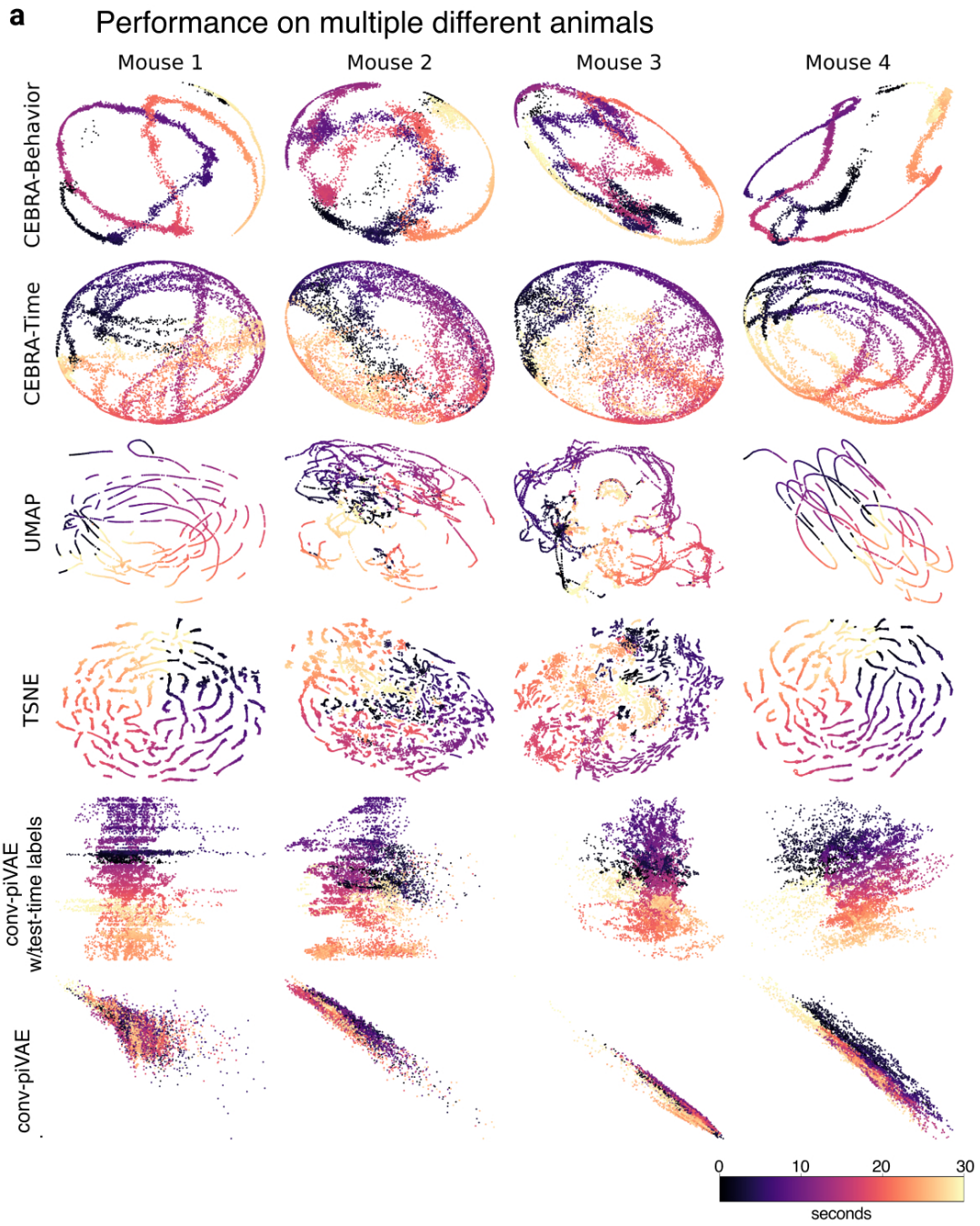
Extended Data Fig. S6. Persistence across dimensions (a): For each dimension of CEBRA-Behavior embedding from the rat hippocampus dataset Betti numbers were computed by applying persistent co-homology. The colored dots are lifespans observed in hypothesis based CEBRA-Behavior. To rule out noisy lifespans, we set a threshold (colored diagonal lines) as maximum lifespan based on 500 seeds of shuffled-CEBRA embedding for each dimension. **(b)**: The topology preserving circular coordinates using the first co-cycle from persistent co-homology analysis on the CEBRA embedding of each dimension is shown (see Methods). The colors indicate position and direction of the rat at the corresponding CEBRA embedding points. **(c)**: The radial angle of each embedding point obtained from **(b)** and the corresponding position and direction of the rat.



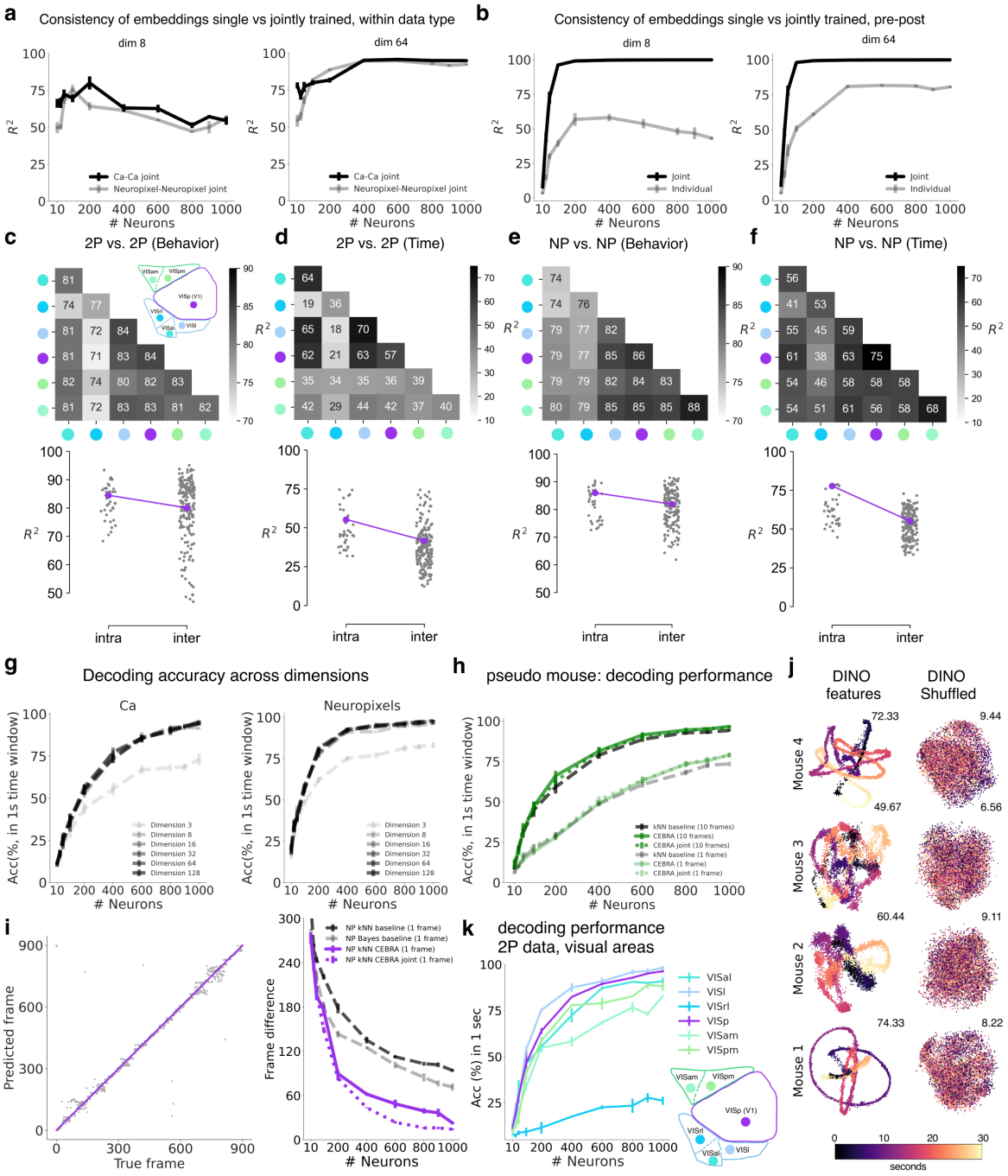
Extended Data Fig. S7. Multi-session training and rapid decoding (a): Top: hippocampus dataset, single animal vs. multi-animal training shows an increase in consistency across animals. Bottom: same for Allen dataset, 4 mice. **(b)**: consistency matrix single vs. multi-session training for hippocampus (32D embedding) and Allen data (128D embedding) respectively. Consistency is reported at the point in training where the average position decoding error is less than 14 cm (corresponds to 7 cm error for rat 1), and a decoding accuracy of 60% on the Allen dataset. **(c)**: Comparison of decoding metrics for single or multi-session training at various consistency levels (averaged across all 12 comparisons). Models were trained for 5,000 (single) or 10,000 (multi-session) steps with a 0.003 learning rate; batch size was 7,200 samples per session. Multi-session training requires longer training or higher learning rates to obtain the same accuracy due to the 4-fold larger batch size, but converges to same decoding accuracy. We plot points at intervals of 500 steps (n=10 seeds); training progresses from lower right to upper left corner within both plots. **(d)**: We demonstrate that we could also adapt to an unseen dataset; here, 3 rats were used for pretraining, and rat #4 was used as a held-out test. The grey lines indicate models trained from scratch (random initialization). We also tested fine-tuning only the input embedding (first layer) or the full model, as the diagram, left, describes. We measured the average time (mean $\pm$ STD) to adapt 100 steps (0.65 $\pm$ 0.13 sec) and 500 steps (3.07 $\pm$ 0.61 sec) on 40 repeated experiments.



Extended Data Fig. S8. Somatosensory cortex decoding from primate recordings (a): We compare CEBRA-Behavior with the cosine similarity and embeddings on the sphere reproduced from Fig. 3b (left) against CEBRA-Behavior trained with the MSE loss and unnormalized embeddings. The embeddings of trials (n=364) of each direction were post-hoc averaged. **(b)**: CEBRA-Behavior trained with x,y position of the hand. Left panel is color-coded to changes in x position and right panel is color-coded to changes in y position. **(c)**: CEBRA-Time without any external behavior variables. As in **b**, left and right are color-coded to x and y position, respectively. **(d)**: Decoding performance of on target direction using CEBRA-Behavior, conv-pi-VAE and a linear classifier. CEBRA-Behavior shows significantly higher decoding performance than the linear classifier (one-way ANOVA, $F(2,75)=3.37$, $p<0.05$ with Post Hoc Tukey HSD $p<0.05$). **(e)**: Loss (InfoNCE) vs. training iteration for CEBRA-Behavior with position, direction, active or passive, and position+direction labels (and shuffled labels) for all trials (left) or only active trials (right), or active trials with a MSE loss. **(f)**: Additional decoding performance results on position and direction-trained CEBRA models with all trial types. For each case, we trained and evaluated 5 seeds represented by black dots and the orange line represents the median.



Extended Data Fig. S9. CEBRA produces consistent, highly decodable embeddings (a): Additional 4 sessions with the most neurons in the Allen visual dataset calcium recording shown for all algorithms we benchmarked (see Methods). For CEBRA-Behavior and CEBRA-Time, we used temperature 1, time offset 10, batch size 128 and 10k training steps. For UMAP, we used a cosine metric and $n\_neighbors$ 15 and min_dist 0.1. For tSNE, we used a cosine metric and $perplexity$ 30. For conv-pi-VAE, we trained with 600 epochs, a batch size of 200 and a learning rate 5×10^{-4}. All methods used 10 time bins input. CEBRA was trained with 3D latent and all other methods were obtained with an equivalent 2D latent dimension.



Extended Data Fig. S10. Spikes and calcium signaling reveal similar embeddings (a): Consistency between the single modality embedding and jointly trained embedding from CEBRA. In higher dimensions, the embedding from single recording modality and the jointly trained embedding became highly consistent with more neurons. **(b)**: Consistency of embeddings from two recording modalities, when a single modality was trained independently and/or jointly trained. The consistency significantly improved with joint training. In higher dimensions, the consistency between single modality embeddings improved as well, which shows that CEBRA can find 'common latents' in two different recording methods (that is theoretically meant to have same information) even without joint training (yet, joint training improves consistency). This data is also presented in Fig. 4e, h, but here plotted together to show improvement with joint training. **(c-f)**: Consistency across modalities and areas for CEBRA-Behavior and -Time (as computed in Fig. 4i-k). The purple dots indicate mean of intra-V1 scores and inter-V1 scores (inter-V1 vs intra-V1 Welch's t-test; 2P (Behavior): $T(10.6)=1.52$, $p=0.081$, 2P (Time): $T(44.3)=4.26$,$p=0.0005$, NP (Behavior): $T(11.6)=2.83$, $p=0.0085$, NP (Time): $T(8.9)=15.51$, $p<0.00001$) **(g)**: CEBRA + kNN decoding performance (see Methods) of CEBRA embeddings of different output embedding dimensions, from calcium (2P) data or Neuropixels, as denoted.**(h)**: Decoding accuracy measured by considering predicted frame being within 1 sec difference to true frame as correct prediction using CEBRA (2P only), jointly trained (2P+NP), or a baseline population vector kNN decoder (using the time window 33 ms (single frame), or 330 ms (10 frame receptive field)). **(i)**: Single frame performance and quantification using CEBRA 1 frame receptive field (NP data), or baseline models. **(j**: As a control experiment we shuffled DINO features: CEBRA-Behavior used the DINO features as behavior labels and CEBRA-Shuffled used the shuffled DINO features. We shuffled the frame order of DINO features within a repeat. Same shuffled order was use for all repeats. Color code is frame number from the movie. The prediction is considered as true if the predicted frame is within 1 sec from the true frame, and the accuracy (%) is noted next to the embedding. For Mice ID 1-4: 337, 353, 397, 475 neurons were recorded, respectively. **(k)**: Decoding performance from 2P data from different visual cortical areas from different layers (2/3, 4, 5/6), as denoted, using a 10 frame window CEBRA-Behavior model using DINO features with 128 output dimension.

Supplementary Information

Suppl. Video 1: "SupplVideo_1.MP4" Corresponding to Fig. 2d. CEBRA-Behavior trained with position+direction on Rat 1. Video is in 2X real-time.

Suppl. Video 2: "SupplVideo_2.MP4" Corresponding to Fig. 5b. The left panels show example calcium traces from 2-photon imaging (top) and spikes from Neuropixels recording (bottom) of primary visual cortex while the video is shown to a mouse. The center panel shows an embedding space constructed by jointly training a CEBRA-Behavior model with 2-photon and Neuropixels recordings using DINO frame features as labels. The trace is embedding of held-out test repeat from Neuropixels recording. The colormap indicates frame number of the 30 second long video (30 Hz). The last panels show true video (top) and the predicted frame sequence (bottom) using kNN decoder on CEBRA-Behavior embedding from the test set. Video is in real-time.

Supplementary Note 1

On identifiability and consistency. When learning (non-linear) representations of a dataset, it is highly desirable that embedding algorithms generate *consistent* embedding spaces. Multiple runs of the algorithm on the same data, multiple runs of the algorithm on data produced in the same way, etc., should generate embedding spaces with a meaningful relation to each other. This "meaningful relation" between algorithm runs can be formalized using tools from identifiability in non-linear independent component analysis (ICA). Suppose we are given two models $\mathbf{f}'$ and $\mathbf{f}^*$ trained on the same dataset, and the performance of these models matches in the sense that they represent the same probability distribution $p' = p^*$. Identifiability then entails that both models are the same up to some known class of transformations (e.g., linear or affine transformations, rotations, permutations and sign-flips, etc.).

For example, one option for parameterizing the distributions is as $p'(\mathbf{y}|\mathbf{x}, \mathbf{y}_1 \dots \mathbf{y}_n) = \exp(\mathbf{f}(\mathbf{x})^\top \mathbf{f}'(\mathbf{y}))/\sum_i \exp(\mathbf{f}(\mathbf{x})^\top \mathbf{f}'(\mathbf{y}_i))$ and respectively for $\tilde{p}'$ defined respectively with $\tilde{\mathbf{f}}$ and $\tilde{\mathbf{f}}'$. Roeder et al. (13) show that if the two distributions match contrastive learning models produce consistent embedding spaces, and it is possible to find a linear mapping $\mathbf{L}$ between the feature spaces, i.e., $\mathbf{L}\mathbf{f}(\mathbf{x}) = \tilde{\mathbf{f}}(\mathbf{x})$ for all $\mathbf{x}$ in the dataset. Other theoretical work has shown that contrastive learning with auxiliary variables is identifiable for bijective neural networks using the noise contrastive estimation (NCE) loss (14), and that with an InfoNCE loss this bijectivity assumption can be removed for certain distributions (25). We will adapt the underlying proofs to our setup in Suppl. Note 2, and give a high-level outline below.

We will consider two important points in both the context of discovery and hypothesis driven training of CEBRA models. Firstly, when applying discovery-driven CEBRA, will two models estimated on comparable experimental data agree in their inferred representation? Second, under which assumptions about the data will we be able able to discover the *true* latent distribution?

Consistency: For consistency across embedding spaces, we require a dataset with a sufficient amount of variability in time. Intuitively, to estimate a d dimensional embedding that is consistent across runs, points sampled from the embedding via the negative distribution q need to vary in at least d directions for each possible reference sample in the dataset. Interestingly, consistency is mostly independent from the data generating process (i.e., the data modality of the recording) and merely requires a sufficiently varying dataset, as well as a choice of feature encoder that passes this variability on to the embedding space.

For example, consider the reaching dataset in Fig. 3 where we showed embeddings that vary in two dimensions (the direction and distance from the center). In this case, the sampling process needs to be designed such that for each reference point we can draw from the dataset, the embedding of the negative samples will vary in at least two directions. This is clearly the case for our training setup: The neurons encode both position and direction information, this information is transformed by the feature encoder, and the resulting embedding varies in at least two directions when the negative distribution samples uniformly across the dataset.

For the first property, we can leverage previous results on the consistency of contrastive learning models over multiple runs (13). Consider the case where we train multiple CEBRA models on data originating from the same data distribution, and consider that we can train these models to full convergence. It is then guaranteed that the embedding spaces will agree up to a linear indeterminacy. In other words, it will always be possible to transform one embedding space into the other by applying a linear transformation. Linear consistency of representations is interesting when we consider linear downstream processing of the inferred embedding space, as is common in neuroscience (1). Such a downstream algorithm (e.g., a linear regression or general linear model) will yield the same performance across different CEBRA models.

Recovering the ground-truth latents: Note that this notion of consistency only makes a statement about the *inferred* latent representation (and identifiability) across multiple runs of the algorithm, but not yet about the relation between this latent representation and the *true* underlying latent variables that generated the data. This is the second property mentioned above, and requires additional assumptions about the data generating process to resolve the ambiguity of what a "latent" underlying a given dataset actually entails. The assumptions concern the injectivity of the data generating process and the positive distribution p. While for discovery driven training, p is an empirical property of the dataset, hypothesis driven training allows to precisely define p based on the observed auxiliary variables. The same theory applies to both cases. Importantly, as for consistency, note that these results are independent from the actual modality of the data and other properties of the signal space we consider. The assumptions are all with respect to the underlying latent distribution.

For the analyses in this paper, the theory for linear identifiability of the underlying latents applies: For time-contrastive training, it is required that the underlying latent distribution is uniform (e.g., there is no inherent bias in the experimental data), and that latents of nearby time steps vary according to a distribution of the form $p(\mathbf{v}|\mathbf{u}) = \exp(\phi(\mathbf{u}, \mathbf{v}))$, where $\mathbf{u}$ and $\mathbf{v}$ underlying the signal variables $\mathbf{x}$ and $\mathbf{y}$. If these conditions are met, the true underlying latents are recovered up to a linear transformation. For hypothesis testing where the user actually specifies the distribution p, this requirement can be easily validated and met.

Relationship between consistency, identifiability, and the sampling mechanism: The CEBRA software package allows for other choices of similarity measures (potentially learnable), which allows to derive guarantees also for these cases. In the most general case, we pick ϕ as a trainable neural network that factorizes into individual components, $\phi(\mathbf{y}, \mathbf{x}) := \sum_i^d \phi_i(y_i, \mathbf{x})$ where each ϕ_i is an individually trained neural network. For sufficiently variable distributions, this allows to recover the underlying latents up to permutations and point-wise non-linear, bijective transformations.

Likewise, it is possible to modify the encoding networks $\mathbf{f}$ and $\mathbf{f}'$ for $\mathbf{x}$ and $\mathbf{y}$, respectively. While our experiments used one network $\mathbf{f} = \mathbf{f}'$ with $\mathbf{x}$ and $\mathbf{y}$ representing neural data, it is well possible to encode different aspects of the dataset and/or to break the symmetry between the two encoding networks and to train a separate network for each of $\mathbf{f}$ and $\mathbf{f}'$. For example, neural data $\mathbf{y}$ could be encoded using $\mathbf{f}'$, and behavior $\mathbf{x}$ could be encoded using $\mathbf{f}$. It would also be possible to use a composition of neural and behavioral data for $\mathbf{x}$. In these cases, if $\mathbf{f}$ and $\mathbf{f}'$ are parameterized as two individual neural networks and ϕ is defined as the dot-product as before, if $\mathbf{y}|\mathbf{x}$ follows a conditionally exponential distribution, we are able to recover all sufficient statistics of this distribution up to a linear transformation.

Examples : As an example of the aforementioned results, let us consider the rat hippocampus dataset used in Fig. 1 and 2. The auxiliary information in this dataset is the position, velocity, and direction of the rat on the linear track.

We can apply different sampling schemes for investigating this dataset. For example, we can apply discovery-driven, time-contrastive learning. In this setup, we sample time steps uniformly from the dataset to arrive at our reference samples. Given a time offset Δ (that informs the algorithm about the time-scale of interest), we obtain positive samples. The resulting batch will be composed of samples $\mathbf{s}_t$ for the reference, $\mathbf{s}(t+\Delta)$ for the positive, and $\mathbf{s}(t_i)$ with uniformly sampled time steps $t_1, \ldots, t_n$ for the negative samples. This corresponds to an approximation of the distribution $p(\mathbf{u}_{t+\Delta}|\mathbf{u}_t)$ of how the latents vary over the course of time. If sufficient variation is present in the dataset along d latent directions, CEBRA models will become, after training, consistent across runs. If additionally the *true* distribution $p(\mathbf{u}_{t+\Delta}|\mathbf{u}_t)$ follows, e.g., a Gaussian distribution, CEBRA will identify the ground truth latents.

In comparison, our so-called hypothesis-driven, behavior-label guided contrastive learning approach would leverage the continuous position information as well as the movement direction of the rat. In the Methods, we denoted the continuous variable as $\mathbf{c}_t$ and the discrete variables as k_t. To arrive at a behavior contrastive embedding using this auxiliary information, we would build a set of differences. If the variables are independent (or should reflect this in the embedding), we build one set $D = \{\mathbf{c}_{t+\Delta} - \mathbf{c}_t\}_{t=1}^T$. For a reference sample at time step t, we sample $\mathbf{d} \sim D$ uniformly, and apply this difference to the position $\mathbf{c}_t$ at step t. We then pick the point closest to $\mathbf{c}_t + \mathbf{d}$ *and* matching the discrete variable k_t as the positive sample.

A lot of variations of this sampling process are possible to embed desirable properties and test hypotheses about the dataset. For instance, consider the primate reaching dataset: Here, $\mathbf{c}_t$ could be selected as the x/y position in space, and the discrete label k_t could denote the reaching direction. However, the 2D differences $\mathbf{c}_{t+\Delta} - \mathbf{c}_t$ will depend on k_t: A reaching direction towards the left will have most variance in negative x-direction $[-1, 0]^\top$, a reaching direction towards the top will have most variance in positive y-direction $[0, 1]^\top$. One way to work around this issue is to consider a polar representation of the position and direction and apply the scheme outlined above. Another alternative is to build the set of differences conditional on the direction k_t, i.e., $D(k) = \{\mathbf{c}_{t+\Delta} - \mathbf{c}_t\}_{t:k_t=k}$. The sampling process is almost analogous to the rat hippocampus example above: We would sample a time step t, look up the discrete variable k_t, but then only sample from $D(k_t)$ to reflect the conditioning on the direction.

Finally, e.g., for very complex movements, it is simple to adapt additional pre-processing schemes. These could involve other deep learning algorithms like DINO used for pre-processing video data in the Allen dataset (to convert pixel data without a

meaningful metric into an embedding space with desirable distance properties); they could also involve simpler processing, such as computing the principal component analysis of a higher dimensional dataset, and using the behavior data in this space.

Many other variations are possible. While the most common use cases are reflected in the CEBRA software toolbox and high-level API and readily usable, more customized use cases can be easily added by the user thanks to a straightforward extension mechanism.

Improving pi-VAE. Zhou et al (5) demonstrate that pi-VAE outperforms LFADS (17), demixed-PCA (61), UMAP (11), PCA, and pfLDS (62) using the rat and/or primate datasets (Extended Data Fig. 1a). We improved the performance of pi-VAE by modifying the encoder, which allows for longer time inputs (Extended Data Fig.1), and this improved version is used throughout, unless noted.

Utilizing CEBRA across contexts. Within our framework, we assume that independent latent variables are combined by a non-linear bijective mixing function to produce neural activity. The latent variables are assumed to change over time, or be correlated to the observed auxiliary variables used to train CEBRA. No additional special structure, or implicit generative models during training are needed.

CEBRA allows for minimizing the impact of selected features on the embedding, while testing the role of others. For example, suppose you have neural data from four different animals, each from the hippocampus while the animal navigated a linear track. You hypothesize that the hippocampus encodes a continuous mapping of space along the track. In this scenario the animal ID is not important, but the spatial location of the animal is. Here, the user can specify to obtain an embedding that is invariant to the animal ID, but should incorporate the position information. Another amendable scenario is a hypothesis-free, discovery-driven approach (akin to unsupervised clustering). Here too, CEBRA can be used, with only time as the input (Fig. 1). Collectively, CEBRA can be used for both visualization of data and latent-space based embedding of neural activity for downstream tasks like decoding.

The flexibility in choosing different auxiliary variables during data analysis allows users to leverage the same algorithm for a variety of applications on a given dataset: Discovery-driven analysis by purely self-supervised learning with time-contrastive learning, hypothesis-driven analysis by comparing embedding quality derived from different behavioral variables, or replacing supervised decoding algorithms, e.g., in brain-machine-interface contexts.

Supplementary Note 2

Here we provide theoretical results for consistency and identifiability of models trained within the CEBRA framework. We proceed by showing properties of the InfoNCE loss (Prop. 1), and use them as the basis for showing that encoders trained on this loss function will become bijective under mild assumptions (Prop. 2). We then revisit existing theory on contrastive learning, and show that CEBRA falls into a category of models for which we can obtain theoretical guarantees on both consistency (Prop 3) across different model runs and identifiability of the ground truth latent distribution for both the discovery-driven (time-contrastive) learning mode (Prop. 6) and the hypothesis-driven mode (Prop. 7). Our results leverage theory by Roeder et al. (13), Hyvärinen et al. (14), Zimmermann et al. (25), Wang and Isola (52).

It should be noted that while consistency results between model runs do not require strong assumptions about the underlying data generating process, understanding the relation between the embedding space given by CEBRA and the underlying *ground truth data generating process* naturally requires such assumptions. However, compared to assumptions in generative models (e.g., VAEs), these assumptions concern the relationship between the ground-truth latent variables, rather than making statements about the signal space.

Notation, data generation, and learning algorithm. We will use the notation presented in the Methods Section. We additionally introduce the latents $\mathbf{u}$ and $\mathbf{v}$ underlying the samples $\mathbf{x}$ and $\mathbf{y}$. We will interchangeably use the distributions $p_{\mathcal{D}}$, p and q for either the latents $\mathbf{u}$ and $\mathbf{v}$ or their respective samples $\mathbf{x}$ and $\mathbf{y}$ depending on their arguments (we will show after Proposition 1 that this treatment is also formally correct due to the training setup considered here). Definitions, propositions and theorems adapted from other works are cited and denoted by upper-case letters and are otherwise adapted to our notation.

Definition 1 (Data generating process and encoder)**.** *Let $\mathbf{u} \in \mathbb{R}^d, \mathbf{v} \in \mathbb{R}^{d'}$ denote latents corresponding to the samples $\mathbf{x} \in \mathbb{R}^D$ and $\mathbf{y} \in \mathbb{R}^{D'}$ in the respective signal space which are generated according to two differentiable and injective mixing functions $\mathbf{g} : \mathbb{R}^d \mapsto \mathbb{R}^D$ and $\mathbf{g}' : \mathbb{R}^{d'} \mapsto \mathbb{R}^{D'}$,*

$$\mathbf{x} = \mathbf{g}(\mathbf{u}), \quad \mathbf{y} = \mathbf{g}'(\mathbf{v}) \tag{4}$$

and there exist optimal differentiable encoders $\mathbf{f} : \mathbb{R}^D \mapsto \mathbb{R}^E$ and $\mathbf{f}' : \mathbb{R}^{D'} \mapsto \mathbb{R}^E$ such that

$$f(\mathbf{g}(\mathbf{u}))_i = u_i \quad f'(\mathbf{g}'(\mathbf{v}))_j = v_j. \tag{5}$$

We will refer to the composition of the data generators and the encoders as $\mathbf{h} = \mathbf{f} \circ \mathbf{g}$ and $\mathbf{h}' = \mathbf{f}' \circ \mathbf{g}'$. In setups where two models are trained on potentially different mixing functions, we denote the second data generator, feature encoder, and the composition of both as $\tilde{\mathbf{h}} = \tilde{\mathbf{f}} \circ \tilde{\mathbf{g}}$ and $\tilde{\mathbf{h}}' = \tilde{\mathbf{f}}' \circ \tilde{\mathbf{g}}'$.

We consider a marginal distribution $p_{\mathcal{D}}(\cdot)$, the positive sample conditional distribution $p(\cdot|\cdot)$ and the negative conditional distribution $q(\cdot|\cdot)$. Reference samples $\mathbf{u}$ from the (true) latent space are mapped to signal space by the injective function $\mathbf{g}$, positive/negative samples $\mathbf{v}$ from the (potentially different latent space) are mapped to (a possibly different) signal space $\mathbf{y}$ by the injective function $\mathbf{g}'$. The encoder $\mathbf{f}$ is applied to $\mathbf{x}$ and the encoder $\mathbf{f}'$ is applied to $\mathbf{y}$ to recover the respective latents underlying $\mathbf{x}$ and $\mathbf{y}$. The similarity measure is denoted as ϕ with $\mathbf{f}(\mathbf{x})$ and $\mathbf{f}'(\mathbf{y})$ as its arguments. Note that ϕ does not need to be a fixed function and can also be parameterized by a learnable neural network. As in the Methods, we use the shortcuts $\psi(\mathbf{x}, \mathbf{y}) = \phi(\mathbf{f}(\mathbf{x}), \mathbf{f}'(\mathbf{y}))$ and additionally introduce $\psi(\mathbf{u}, \mathbf{v}) := \phi(\mathbf{h}(\mathbf{u}), \mathbf{h}'(\mathbf{v}))$ without additional subscripts, as the desired shortcut will be clear from the context and its arguments.

Note that this is a very general setup. We typically would assume that the number of dimensions in the (shared) latent space is the same, $d = d'$, and could further assume that the number of dimensions E of the embedding space is also matching.

We recall the contrastive objective in the limit of unlimited samples from the main text:

Definition 2 (Generalized InfoNCE objective)**.** *In the limit of unlimited negative samples, the InfoNCE objective is a functional*

$$\mathcal{L}[\psi]_{\text{asympt}} = \int p_{\mathcal{D}}(\mathbf{x}) \left[\log \int q(\mathbf{y}|\mathbf{x}) e^{\psi(\mathbf{x},\mathbf{y})} d\mathbf{y} - \int p(\mathbf{y}|\mathbf{x}) \psi(\mathbf{x}, \mathbf{y}) d\mathbf{y} \right] d\mathbf{x}, \tag{6}$$

depending on the positive sample conditional density $p(\mathbf{y}|\mathbf{x})$, the negative sample density $q(\mathbf{y}|\mathbf{x})$, the marginal density $p_{\mathcal{D}}(\mathbf{x})$ and the embedding similarity ψ as defined above.

We call this objective "generalized" as it extends the original definition of the InfoNCE loss used in the literature. Oord et al. (20) introduced an objective where the marginal (there: prior) $p_{\mathcal{D}}$ and the negative sample distribution q matched, which influences the types of functions that can be learned. The discussion of the objective by Wang and Isola (52) makes stronger

assumptions about the nature of the conditional distribution p for the positive pair, and only considers uniform choices for the marginal $p_{\mathcal{D}}$ and negative conditional q.

Overall, in CEBRA, the key difference to prior uses of the InfoNCE objective is the ability to control the properties of the embedding space through p and q and ϕ, and leverage this to retrieve the discovery-driven, hypothesis-driven, and hybrid modes as demonstrated in the main text. In fact, for hypothesis-driven training, the distributions used for sampling do not even need to be connected to the underlying data generating process—instead, varying p and testing to enforce various neighbourhood relations on the neural data is used as a tool to discover meaningful relations between the auxiliary variables (e.g., behavior) and signal (e.g., neural activity). In the same manner, p and q can be selected such that a particular factor is sampled uniformly, to enforce invariance (e.g. across a subject, or a modality variable).

The loss optimized in practice acts on a limited number of negative samples in each mini-batch:

Definition 3 (Generalized InfoNCE objective with limited batch size)**.** *For a fixed number of negative samples n, the InfoNCE objective is the functional*

$$\mathcal{L}[\psi]_n = \mathop{\mathbb{E}}_{\substack{\mathbf{x}\sim p_{\mathcal{D}}(\mathbf{x}),\ \mathbf{y}_+\sim p(\mathbf{y}|\mathbf{x}) \\ \mathbf{y}_1,\dots,\mathbf{y}_n\sim q(\mathbf{y}|\mathbf{x})}} \left[-\psi(\mathbf{x},\mathbf{y}_+) + \log \sum_{i=1}^{n} e^{\psi(\mathbf{x},\mathbf{y}_i)} \right],$$

depending on the positive sample conditional density $p(\mathbf{y}|\mathbf{x})$, the negative sample density $q(\mathbf{y}|\mathbf{x})$, the marginal density $p_{\mathcal{D}}(\mathbf{x})$ and the embedding similarity ψ as defined above.

Both losses can be related due to Theorem 1 by Wang and Isola (52). In the limit of unlimited samples $n \to \infty$, we obtain for the batch size n:

$$\mathcal{L}[\psi]_{\text{asympt}} = \lim_{n\to\infty} \left(\mathcal{L}[\psi]_n - \log n\right). \tag{7}$$

For a sufficiently large batch size, we can leverage the quantity $\mathcal{L}[\psi]_n - \log n$ as a goodness of fit measure (as outlined in the Methods) that estimates the distance from a "default" embedding. When comparing models with equal batch size n, note that the InfoNCE loss can also directly serve as this metric.

Minimizers of the generalized InfoNCE loss. In this section, we show that optimizing the generalized InfoNCE objective from Def. 2 yields the unique minimizer $\psi(\mathbf{x},\mathbf{y}) = C(\mathbf{x}) + \log p(\mathbf{y}|\mathbf{x})/q(\mathbf{y}|\mathbf{x})$ or equivalently $\psi(\mathbf{u},\mathbf{v}) = C'(\mathbf{u}) + \log p(\mathbf{v}|\mathbf{u})/q(\mathbf{v}|\mathbf{u})$ up to an arbitrary constant function $C'(\mathbf{u})$ than can depend on the latents of the reference latents. In this regard, the InfoNCE loss is more flexible than the standard noise constrastive estimation (NCE) loss which has a similar minimizer, but is limited to $C(\mathbf{x}) = C'(\mathbf{u}) = 0$. The minimum loss value is the negative Kullbach-Leibler divergence between the positive and negative distributions. To obtain non-trivial solutions, it is therefore important that p and q differ in a non-trivial way (which is the case for both time-contrastive and behavior-contrastive sampling outlined in the context of CEBRA).

Proposition 1. *Let $p(\cdot|\cdot)$ be the conditional distribution of the positive samples, $q(\cdot|\cdot)$ the conditional distribution of the negative samples and $p_{\mathcal{D}}(\cdot)$ the marginal distribution of the reference samples. The generalized InfoNCE objective (Def. 2) is convex in ψ with the unique minimizer*

$$\psi^*(\mathbf{x},\mathbf{y}) = \log \frac{p(\mathbf{y}|\mathbf{x})}{q(\mathbf{y}|\mathbf{x})} + C(\mathbf{x}), \quad \textit{with} \quad \mathcal{L}[\psi^*]_{\text{asympt}} = -\mathcal{D}_{\text{KL}}(p(\cdot|\cdot)\|q(\cdot|\cdot)) \tag{8}$$

on the support of $p_{\mathcal{D}}$, where $C : \mathbb{R}^d \to \mathbb{R}$ is an arbitrary mapping.

Proof. We rewrite the objective as

$$\mathcal{L}[\psi]_{\text{asympt}} = \int p_{\mathcal{D}}(\mathbf{x}) \left[\log \int q(\mathbf{y}|\mathbf{x}) e^{\psi(\mathbf{x},\mathbf{y})} d\mathbf{y} - \int p(\mathbf{y}|\mathbf{x})\psi(\mathbf{x},\mathbf{y}) d\mathbf{y} \right] d\mathbf{x}, \tag{9}$$

and we can compute the first-order functional derivative (using the method discussed in Cahill 2014[1])

$$\begin{aligned} \delta\mathcal{L}[\psi][h]_{\text{asympt}} &= \frac{d}{d\epsilon}\mathcal{L}[\psi+\epsilon h]\big|_{\epsilon=0} \\ &= \int p_{\mathcal{D}}(\mathbf{x}) \left[\frac{1}{Z_\psi(\mathbf{x})} \int q(\mathbf{y}|\mathbf{x}) e^{\psi(\mathbf{x},\mathbf{y})} h(\mathbf{x},\mathbf{y}) d\mathbf{y} - \int p(\mathbf{y}|\mathbf{x}) h(\mathbf{x},\mathbf{y}) d\mathbf{y} \right] d\mathbf{x}, \\ &\quad \text{with } Z_\psi(\mathbf{x}) = \int q(\mathbf{y}'|\mathbf{x}) e^{\psi(\mathbf{x},\mathbf{y}')} d\mathbf{y}'. \end{aligned} \tag{10}$$

[1] http://quantum.phys.unm.edu/523-14/ch15.pdf

The first-order functional derivative vanishes for all functions $h(\mathbf{x}, \mathbf{y})$ whenever

$$p_{\mathcal{D}}(\mathbf{x}) \left[\frac{1}{Z_\psi(\mathbf{x})} q(\mathbf{y}|\mathbf{x}) e^{\psi(\mathbf{x},\mathbf{y})} - p(\mathbf{y}|\mathbf{x}) \right] = 0. \tag{11}$$

This is the case iff at any point $(\mathbf{x}, \mathbf{y})$, either $p_{\mathcal{D}}(\mathbf{x}) = 0$ or

$$Z_\psi(\mathbf{x}) = \frac{q(\mathbf{y}|\mathbf{x})}{p(\mathbf{y}|\mathbf{x})} e^{\psi(\mathbf{x},\mathbf{y})}. \tag{12}$$

Since the left hand side of Eq. Eq. (12) is independent of $\mathbf{y}$, the right hand side must be independent of $\mathbf{y}$ as well. Hence all functions $\psi^*(\mathbf{x}, \mathbf{y})$ which are solutions to Eq. Eq. (12) are of the form

$$\psi^*(\mathbf{x}, \mathbf{y}) = \log \frac{p(\mathbf{y}|\mathbf{x})}{q(\mathbf{y}|\mathbf{x})} + C(\mathbf{x}), \tag{13}$$

where C is an arbitrary function depending only on $\mathbf{x}$ and not on $\mathbf{y}$. Then by definition of $Z_\psi(\mathbf{x})$,

$$Z_\psi(\mathbf{x}) = \int q(\mathbf{y}'|\mathbf{x}) e^{\psi(\mathbf{x},\mathbf{y}')} d\mathbf{y}' = e^{C(\mathbf{x})} \tag{14}$$

which is consistent when inserted into Eq. Eq. (12). Therefore the minimizers of $\mathcal{L}[\psi]_{\text{asympt}}$ form a convex connected set $\mathcal{M}$,

$$\mathcal{M} = \left\{ \psi^* : \psi^*(\mathbf{x}, \mathbf{y}) = \begin{cases} \log \frac{p(\mathbf{y}|\mathbf{x})}{q(\mathbf{y}|\mathbf{x})} + C(\mathbf{x}) & \text{if } p_{\mathcal{D}}(\mathbf{x}) \neq 0 \\ f(\mathbf{x}, \mathbf{y}) & \text{if } p_{\mathcal{D}}(\mathbf{x}) = 0 \end{cases} \right\}, \tag{15}$$

where C, f are arbitrary functions. It can be checked that all minima achieve the same value $L[\psi^*]$ given by

$$\mathcal{L}[\psi^*]_{\text{asympt}} = -\int p_{\mathcal{D}}(\mathbf{x}) \int p(\mathbf{y}|\mathbf{x}) \log \frac{p(\mathbf{y}|\mathbf{x})}{q(\mathbf{y}|\mathbf{x})} d\mathbf{y} = -\int p_{\mathcal{D}}(\mathbf{x}) D_{\text{KL}} \left[p(\cdot|\mathbf{x}) || q(\cdot|\mathbf{x}) \right] d\mathbf{x} \leq 0. \tag{16}$$

It is left to show that the objective function is convex. The second-order functional derivative is given by

$$\begin{aligned} \delta^2 \mathcal{L}[\psi][h]_{\text{asympt}} =& \frac{d^2}{d\epsilon^2} \mathcal{L}[\psi + \epsilon h]\big|_{\epsilon=0} \\ =& \frac{d}{d\epsilon} \int p(\mathbf{x}) \left[\frac{1}{Z_{\psi+\epsilon h}(\mathbf{x})} \int q(\mathbf{y}|\mathbf{x}) e^{\psi(\mathbf{x},\mathbf{y}) + \epsilon h(\mathbf{x},\mathbf{y})} h(\mathbf{x}, \mathbf{y}) d\mathbf{y} - \int p(\mathbf{y}|\mathbf{x}) h(\mathbf{x}, \mathbf{y}) d\mathbf{y} \right] \Bigg|_{\epsilon=0} \\ =& \int p(\mathbf{x}) \left[\mathbb{E}_{g(\mathbf{y}|\mathbf{x})} \left[h(\mathbf{x}, \mathbf{y})^2 \right] - \mathbb{E}_{g(\mathbf{y}|\mathbf{x})} \left[h(\mathbf{x}, \mathbf{y}) \right]^2 \right], \\ & \text{with } g(\mathbf{y}|\mathbf{x}) = \frac{1}{Z_\psi(\mathbf{x})} q(\mathbf{y}|\mathbf{x}) e^{\psi(\mathbf{x},\mathbf{y})}, \text{ and } \int g(\mathbf{y}|\mathbf{x}) d\mathbf{y} = 1. \end{aligned} \tag{17}$$

Since $g(\mathbf{y}|\mathbf{x})$ is a probability density function, we can apply Jensen's inequality to the convex function $A \to A^2$ for the random variable $A := h(\mathbf{x}, \mathbf{y})$ to obtain

$$\mathbb{E}_{g(\mathbf{y}|\mathbf{x})} \left[h(\mathbf{x}, \mathbf{y})^2 \right] - \left(\mathbb{E}_{g(\mathbf{y}|\mathbf{x})} \left[h(\mathbf{x}, \mathbf{y}) \right] \right)^2 \geq 0 \quad \Rightarrow \delta^2 \mathcal{L}[\psi][h]_{\text{asympt}} = \int p(\mathbf{x}) \left[\mathbb{E}_{g(\mathbf{y}|\mathbf{x})} \left[h(\mathbf{x}, \mathbf{y})^2 \right] - \left(\mathbb{E}_{g(\mathbf{y}|\mathbf{x})} \left[h(\mathbf{x}, \mathbf{y}) \right] \right)^2 \right] \geq 0. \tag{18}$$

and it follows that the InfoNCE loss is convex in ψ.

To prove uniqueness of the minimum: Note that since the mapping $A \to A^2$ is not affine, a necessary and sufficient condition for equality is A to be constant, which holds iff $h(\mathbf{x}, \mathbf{y}) = h(\mathbf{x}, \mathbf{y}') := h(\mathbf{x})$ for all $\mathbf{y}$. The objective is hence strictly convex for all variations involving a variation in $\mathbf{y}$, and the second derivative vanishes for variations that only depend on $\mathbf{x}$. Variations that only depend on $\mathbf{x}$ are represented by the function C which appeared in the set of minimizers $\mathcal{M}$. □

Note that the difference between the minimizer of the NCE loss and the InfoNCE loss is the additional constant function C depending on the reference sample, which makes the loss function more flexible. Another common formulation of the InfoNCE minimizer in the literature is given as $\psi(\mathbf{x}, \mathbf{y}) = \log p(\mathbf{x}, \mathbf{y}) / (p_{\mathcal{D}}(\mathbf{x}) p_{\mathcal{D}}(\mathbf{y}))$ which is a special case of our more general solution if $C(\mathbf{x}) = -\log p_{\mathcal{D}}(\mathbf{x})$, the negative distribution is chosen to be the marginal, $q = p_{\mathcal{D}}$, and the learning setup is symmetric.

Let us also confirm that our interchangeable use of the latents $(\mathbf{u}, \mathbf{v})$ and signal variables $(\mathbf{x}, \mathbf{y})$ is formally correct; due to the transformation theorem for any distribution $p_{\mathbf{u}}(\mathbf{u}) = p_{\mathbf{x}}(\mathbf{g}(\mathbf{u})) \det \mathbf{J}_{\mathbf{g}}(\mathbf{u})$ and respectively for $\mathbf{v}, \mathbf{g}', \mathbf{y}$. At the minimizer, we then arrive at

$$\log \frac{p(\mathbf{y}|\mathbf{x})}{q(\mathbf{y}|\mathbf{x})} + C(\mathbf{x}) = \log \frac{p(\mathbf{y}|\mathbf{x}) p_{\mathcal{D}}(\mathbf{x})}{q(\mathbf{y}|\mathbf{x}) p_{\mathcal{D}}(\mathbf{x})} + C(\mathbf{x}) \tag{19}$$

$$= \log \frac{p(\mathbf{v}|\mathbf{u}) p_{\mathcal{D}}(\mathbf{u}) \det \mathbf{J}_{\mathbf{g}}(\mathbf{u}) \det \mathbf{J}_{\mathbf{g}'}(\mathbf{v})}{q(\mathbf{v}|\mathbf{u}) p_{\mathcal{D}}(\mathbf{u}) \det \mathbf{J}_{\mathbf{g}}(\mathbf{u}) \det \mathbf{J}_{\mathbf{g}'}(\mathbf{v})} + C'(\mathbf{u}) \tag{20}$$

$$= \log \frac{p(\mathbf{v}|\mathbf{u})}{q(\mathbf{v}|\mathbf{u})} + C'(\mathbf{u}), \quad \text{with } C(\mathbf{g}(\mathbf{u})) = C'(\mathbf{u}), \tag{21}$$

i.e., the minimizer can be equivalently written in terms of the latents and the signal variables.

Minimizers of the InfoNCE loss become bijective. A property that allows us to weaken some of the conditions given by Hyvärinen et al. (14), Zimmermann et al. (25) and Roeder et al. (13) is the observation that the composition of data generating process and feature encoder becomes bijective for the optimal value of the generalized InfoNCE objective. We introduce the following:

Definition 4 (Diversity condition for bijectivity)**.** *The sampling process composed of distributions p and q is sufficiently diverse if their log-likelihoods satisfy*

$$\operatorname{rank}\left(\left[\frac{\partial^2 \log p(\mathbf{v}|\mathbf{u})}{\partial u_i \partial v_j} \right]_{i \in [d], j \in [d]} - \left[\frac{\partial^2 \log q(\mathbf{v}|\mathbf{u})}{\partial u_i \partial v_j} \right]_{i \in [d], j \in [d]} \right) = d, \tag{22}$$

for all $\mathbf{u}$ in the support of the marginal distribution $p_{\mathcal{D}}$ and $d = d'$.

Def. 4 is a mild condition on the distributions p and q: Intuitively, the condition requires that for all samples $\mathbf{u}$, we can sample positive samples $\mathbf{v}$ that sufficiently vary in all d latent directions, which would be independent from the samples given by the negative distribution q. Suppose q is chosen to be uniform; then the condition is fulfilled for common choices like a Normal distribution with $\log p(\mathbf{v}|\mathbf{u}) = Z(\mathbf{u}) - (\mathbf{u} - \mathbf{v})^\top \mathbf{\Sigma} (\mathbf{u} - \mathbf{v})$ (where $\operatorname{rank}(-\mathbf{\Sigma}) = d$) or a von Mises-Fisher distribution with $\log p(\mathbf{v}|\mathbf{u}) = Z(\mathbf{u}) + \kappa \mathbf{u}^\top \mathbf{v}$ (where $\operatorname{rank} \kappa \mathbf{I} = d$).

We make two additional observations: Firstly, for simple distributions q that do not depend on $\mathbf{u}$, the diversity assumption only affects the positive distribution p as the second term vanishes. Secondly, if p and q are selected to train the network to become invariant to one factor v_i with $p(\mathbf{v}|\mathbf{u}) = p(v_i) p(\mathbf{v}_i|\mathbf{u})$ and $q(\mathbf{v}|\mathbf{u}) = p(v_i) q(\mathbf{v}_i|\mathbf{u})$, the distributions $p(v_i)$ will cancel out in the condition, and reduce the rank by one dimension (which is as intended, as the factor should be discarded during training).

From this diversity condition, we can derive bijectivity of the composition $\mathbf{h} = \mathbf{f} \circ \mathbf{g}$ of the data generating process and feature encoder:

Proposition 2. *Assume that:*

1. *ψ with $\psi(\mathbf{u}, \mathbf{v}) = \phi(\mathbf{h}(\mathbf{u}), \mathbf{h}'(\mathbf{v}))$ is a minimizer of the InfoNCE objective (Def. 2) in a learning setup as outlined in Def 1.*
2. *The distributions p and q satisfy the diversity condition for bijectivity (Def. 4).*

Then $\mathbf{h}$ and $\mathbf{h}'$ are bijective on the support of $p_{\mathcal{D}}$.

Proof. By Proposition 1, the minimizer of the InfoNCE loss on the support of $p_{\mathcal{D}}$ is

$$\psi(\mathbf{u}, \mathbf{v}) = \log \frac{p(\mathbf{v}|\mathbf{u})}{q(\mathbf{v}|\mathbf{u})} + C(\mathbf{u}) \tag{23}$$

For ψ, we compute the second derivatives and arrange them in matrix form as

$$\left[\frac{\partial^2 \psi(\mathbf{u}, \mathbf{v})}{\partial u_i \partial v_j} \right]_{i \in [d], j \in [d]} = \mathbf{J}^\top(\mathbf{u}) \mathbf{P}(\mathbf{h}(\mathbf{u}), \mathbf{h}'(\mathbf{v})) \mathbf{J}'(\mathbf{v}) \tag{24}$$

where we used the shorthand $\mathbf{P}(\mathbf{a},\mathbf{b})_{ij} := \partial^2\phi(\mathbf{a},\mathbf{b})/\partial\mathbf{a}_i\mathbf{b}_j$. $\mathbf{J}$ is the Jacobi matrix of $\mathbf{h}$, and $\mathbf{J}'$ is the Jacobi matrix of $\mathbf{h}'$. For the right hand side of the previous equation, we note that

$$\operatorname{rank}(\mathbf{J}^\top(\mathbf{u})\mathbf{P}(\mathbf{h}(\mathbf{u}),\mathbf{h}'(\mathbf{v}))\mathbf{J}'(\mathbf{v})) \leq \min\{\operatorname{rank}\mathbf{J}(\mathbf{u}),\operatorname{rank}\mathbf{J}'(\mathbf{v}),\operatorname{rank}\mathbf{P}(\mathbf{h}(\mathbf{u}),\mathbf{h}'(\mathbf{v}))\}, \tag{25}$$

and the rank of the left hand side is given by inserting Eq. 23 into the diversity assumption (2):

$$\operatorname{rank}\left[\frac{\partial^2\psi(\mathbf{u},\mathbf{v})}{\partial u_i \partial v_j}\right]_{i\in[d],j\in[d]} = \operatorname{rank}\left(\left[\frac{\partial^2(\log p(\mathbf{v}|\mathbf{u}) - \log q(\mathbf{v}|\mathbf{u}) + C(\mathbf{u}))}{\partial u_i \partial v_j}\right]_{i\in[d],j\in[d]}\right) = d.$$

Combining both results gives

$$\operatorname{rank}(\mathbf{J}^\top(\mathbf{u})\mathbf{P}(\mathbf{h}(\mathbf{u}),\mathbf{h}'(\mathbf{v}))\mathbf{J}'(\mathbf{v})) = d \leq \min\{\operatorname{rank}\mathbf{J}(\mathbf{u}),\operatorname{rank}\mathbf{J}'(\mathbf{v}),\operatorname{rank}\mathbf{P}(\mathbf{h}(\mathbf{u}),\mathbf{h}'(\mathbf{v}))\}, \tag{26}$$

and implies

$$\operatorname{rank}\mathbf{J}(\mathbf{v}) = \operatorname{rank}\mathbf{J}'(\mathbf{u}) = \operatorname{rank}\mathbf{P}(\mathbf{h}(\mathbf{u}),\mathbf{h}'(\mathbf{v})) = d. \tag{27}$$

Then, both Jacobi matrices have full rank on the support of $p_{\mathcal{D}}$, hence $\mathbf{h}$ and $\mathbf{h}'$ are bijective, concluding the proof. □

Notably, this result is independent of the particular choice of the (potentially learnable) similarity measure ϕ. The similarity measure is implicitly constrained by the requirement that ψ needs to match the log-likelihood ratio of p and q up to a constant.

CEBRA models are consistent. We proceed by showing that CEBRA models are *consistent* under weak assumptions on the data distribution. Consistency entails that the embedding spaces of two different models can be mapped onto each other by some known transformation. In this subsection, we consider the class of *linear* transformations and in the following subsection we will discuss alternative transformations. We denote two independently trained CEBRA models as $\{\mathbf{f},\mathbf{f}'\}$ and $\{\tilde{\mathbf{f}},\tilde{\mathbf{f}}'\}$, and make statements about when linear transformations exist such that $\mathbf{f} = \mathbf{L}\tilde{\mathbf{f}}$ and $\mathbf{f}' = \mathbf{M}\tilde{\mathbf{f}}'$ for two full rank matrices $\mathbf{L}$ and $\mathbf{M}$.

We begin by recalling the Canonical Discriminative Form and Diversity Condition in Roeder et al. (13), adapted to our notation:

Definition A (Canonical Discriminative Form, Roeder et al. (13))**.** *Given a data distribution $p_{\mathcal{D}}(\mathbf{x},\mathbf{y},S)$ with random variables $\mathbf{x}$ and $\mathbf{y}$ and a set S containing the possible values of $\mathbf{y}$ given $\mathbf{x}$,*

$$p_{\mathcal{D}}(\mathbf{y}|\mathbf{x},S) > 0 \iff \mathbf{y} \in S, \tag{28}$$

a generalized discriminative model family may be defined by its parameterization of the probability of the target variable $\mathbf{y}$ conditioned on an observed variable $\mathbf{x}$ and the set S that contains not only the true target label $\mathbf{y}$, but also a collection of distractors $\mathbf{y}'$:

$$p_{\mathbf{f},\mathbf{f}'}(\mathbf{y}|\mathbf{x},S) = \frac{\exp(\mathbf{f}(\mathbf{x})^\top\mathbf{f}'(\mathbf{y}))}{\sum_{\mathbf{y}'\in S}\exp(\mathbf{f}(\mathbf{x})^\top\mathbf{f}'(\mathbf{y}'))}. \tag{29}$$

Note that the feature extractors $\mathbf{f}$ and $\mathbf{f}'$ could be two separate networks, as we already discussed in Suppl. Note 1. Roeder et al. (13) consider $\mathbf{f}$ to be a "data encoder" and $\mathbf{f}'$ to be a "context encoder". This is in contrast to the setup by Hyvärinen et al. (14) which we will revisit in the context of recovering the data generating factors, where $\mathbf{f}(\mathbf{x})$ would play the role of an auxiliary variable while $\mathbf{f}'$ is the feature encoder later used in downstream tasks and analysis.

In CEBRA, the choice and role of both functions can be configured, which is why we will discuss theoretical guarantees for both use cases. We proceed by re-stating two diversity conditions needed for $\mathbf{f}$ or $\mathbf{f}'$ to become consistent:

Definition B (Diversity conditions for consistency, Roeder et al. (13))**.** *Let $Z(\mathbf{x},S) := -\log\sum_{\mathbf{y}'\in S}\exp(\mathbf{f}(\mathbf{x})^\top\mathbf{f}'(\mathbf{y}'))$. Assume that for the encoders $(\mathbf{f},\mathbf{f}',\tilde{\mathbf{f}},\tilde{\mathbf{f}}')$ for which it holds that $p_{\mathbf{f},\mathbf{f}'} = p_{\tilde{\mathbf{f}},\tilde{\mathbf{f}}'}$*

1. *for any given $\mathbf{y}$, there exist $M+1$ tuples $\{(\mathbf{x}^{(i)},S^{(i)})\}_{i=1}^{M+1}$, such that $p_{\mathcal{D}}(\mathbf{x}^{(i)},\mathbf{y},S^{(i)}) > 0$, and such that the $((M+1)\times(M+1))$ matrices $\mathbf{M}$ and $\tilde{\mathbf{M}}$ are invertible, where $\mathbf{M}$ consists of columns $[-Z(\mathbf{x}^{(i)},S^{(i)});\mathbf{f}(\mathbf{x}^{(i)})]$, and $\tilde{\mathbf{M}}$ consists of columns $[-Z(\mathbf{x}^{(i)},S^{(i)});\tilde{\mathbf{f}}(\mathbf{x}^{(i)})]$,*
2. *for any given $\mathbf{x}$, by repeated sampling $S\sim p_D(S|\mathbf{x})$ and picking two points $\mathbf{y}_A,\mathbf{y}_B \in S$, we can construct a set of M distinct tuples $(\mathbf{y}_A^{(i)},\mathbf{y}_B^{(i)})_{i=1}^{M}$ such that the matrices $\mathbf{L}$ and $\tilde{\mathbf{L}}$ are invertible, where $\mathbf{L}$ consists of columns $(\mathbf{f}'(\mathbf{y}_A(i)) - \mathbf{f}'(\mathbf{y}_B(i)))$, and $\tilde{\mathbf{L}}$ consists of columns $(\tilde{\mathbf{f}}'(\mathbf{y}_A(i)) - \tilde{\mathbf{f}}'(\mathbf{y}_B(i))$, $i\in 1,\ldots,M$.*

With the diversity condition in place, we recall Theorem 1 from Roeder et al. (13), adapted to our notation:

Theorem A (Roeder et al. (13))**.** *Under the diversity condition (Def. B), models following the canonical discriminative form (Def. A) are linearly identifiable. That is, for any encoders* $\{\mathbf{f}, \mathbf{f}'\}$*, and* $\{\tilde{\mathbf{f}}, \tilde{\mathbf{f}}'\}$ *it holds that*

$$p_{\mathbf{f},\mathbf{f}'} = p_{\tilde{\mathbf{f}},\tilde{\mathbf{f}}'} \quad \Longrightarrow \quad \mathbf{f}(\mathbf{x}) = \mathbf{L}\tilde{\mathbf{f}}(\mathbf{x}), \quad \mathbf{f}'(\mathbf{y}) = \mathbf{L}\tilde{\mathbf{f}}'(\mathbf{y}) \tag{30}$$

for all samples ($\mathbf{x}$*,* $\mathbf{y}$*) in the support of the data distribution.*

Proof. See Theorem 1, Roeder et al. (13), where we replaced the equivalence condition on the right hand side by inserting its definition for clarity. □

We can leverage this result as CEBRA falls into the class of models models described in Definition A:

Proposition 3 (CEBRA models are consistent.)**.** *Assume that two CEBRA models are trained on data from the same latent data distribution, and denote the feature encoders of the trained models as* $\mathbf{f}, \mathbf{f}'$ *and* $\tilde{\mathbf{f}}, \tilde{\mathbf{f}}'$*. Further assume that for both models, the similarity measure* ϕ *is the dot-product similarity and* ψ *minimizes the generalized InfoNCE loss. Finally assume that the diversity condition (Def. B) holds. Then the feature encoders* $\mathbf{f}$*,* $\mathbf{f}'$ *are consistent up to a linear transformation, and* $\mathbf{f}(\mathbf{x}) = \mathbf{L}\tilde{\mathbf{f}}(\mathbf{x})$*,* $\mathbf{f}'(\mathbf{y}) = \mathbf{L}'\tilde{\mathbf{f}}'(\mathbf{y})$*, for linear transformations* $\mathbf{L}$*,* $\mathbf{L}'$ *for any pair of points* $\mathbf{x}, \mathbf{y}$ *in the support of the data distribution.*

Proof. The generalized InfoNCE objective with limited samples (Def. 3) can be written as

$$\mathop{\mathbb{E}}_{\substack{\mathbf{x}\sim p(\mathbf{x}),\ \mathbf{y}_+\sim p(\mathbf{y}|\mathbf{x}) \\ \mathbf{y}_1,\dots,\mathbf{y}_n\sim q(\mathbf{y}|\mathbf{x})}} [\log p(\mathbf{y}^+|\mathbf{x}, S)], \quad \log p(\mathbf{y}^+|\mathbf{x}, S) = -\log \frac{\exp \psi(\mathbf{x}, \mathbf{y}_+)}{\sum_{i=1}^n \exp \psi(\mathbf{x}, \mathbf{y}_i)} = -\log \frac{\exp(\mathbf{f}(\mathbf{x})^\top \mathbf{f}'(\mathbf{y}_+)/\tau)}{\sum_{i=1}^n \exp(\mathbf{f}(\mathbf{x})^\top \mathbf{f}'(\mathbf{y}_i)/\tau)} \tag{31}$$

which matches the canonical discriminative form (Def. A) with encoders $\mathbf{f}(\cdot)$ and $\mathbf{f}'(\cdot)/\tau$. The composition of the set $S := \{\mathbf{y}_i\}_{i=1}^N$ is given by the distribution $p_{\mathcal{D}}(\mathbf{x})q(\mathbf{y}|\mathbf{x})$ and S fulfills Def. B by assumption. At the minimizer, the values of the loss functions match, from which it follows that $p_{\mathbf{f},\mathbf{f}'} = p_{\tilde{\mathbf{f}},\tilde{\mathbf{f}}'}$. Hence, we apply Theorem A to find that $\mathbf{f}$ and $\mathbf{f}'$ are consistent up to a linear transform, concluding the proof. □

It is worth noting that this result also *holds for datasets with limited samples*, i.e., the objective in Def. 3, as long as the dataset fulfills the diversity condition (Def. B). Checking the diversity condition as well as matching distributions for the two CEBRA models is possible in practice using only the dataset and the trained model.

Both diversity conditions from Roeder et al. (13) depend on the variability of the ground truth latent distribution (and the presence of this variability after mapping the latents to signal space), and on the properties of the encoders $\mathbf{f}$ and $\mathbf{f}'$. While Roeder et al. (13) already discuss that in practice, a randomized neural network will fulfill the diversity conditions, with our diversity criterion for bijectivity (Def. 4) and the bijectivity of $\mathbf{f}$ and $\mathbf{f}'$ that follows, we can strengthen this argument and ensure that both conditions hold upon convergence for minimizers of the generalized InfoNCE objective:

Proposition 4. *Assume that the encoders* $(\mathbf{f}, \mathbf{f}', \tilde{\mathbf{f}}, \tilde{\mathbf{f}}')$ *and hence also the compositions of data generator and encoders* $(\mathbf{h}, \mathbf{h}', \tilde{\mathbf{h}}, \tilde{\mathbf{h}}')$ *minimize the InfoNCE objective. Assume that upon convergence, the diversity condition for bijectivity (Def. 4) holds. Then,* $\mathbf{h}(\mathbf{u}) = \mathbf{A}\mathbf{h}(\mathbf{u})$ *and* $\mathbf{h}'(\mathbf{v}) = \mathbf{B}\tilde{\mathbf{h}}'(\mathbf{v})$ *for all latents* $\mathbf{u}, \mathbf{v}$ *in the support of the data distribution for two full-rank matrices* $\mathbf{A}$*,* $\mathbf{B}$*.*

Proof. Both models share the minimizer

$$\mathbf{h}(\mathbf{u})^\top \mathbf{h}'(\mathbf{v}) = \tilde{\mathbf{h}}(\mathbf{u})^\top \tilde{\mathbf{h}}'(\mathbf{v}) = \log \frac{p(\mathbf{v}|\mathbf{u})}{q(\mathbf{v}|\mathbf{u})} + C(\mathbf{u}) \tag{32}$$

and we have

$$\mathbf{h}(\mathbf{u})^\top \mathbf{h}'(\mathbf{v}) = \tilde{\mathbf{h}}(\mathbf{u})^\top \tilde{\mathbf{h}}'(\mathbf{v}) \tag{33}$$

taking derivatives with respect to $\mathbf{v}$, and then to $\mathbf{u}$, we arrive at

$$\mathbf{J}(\mathbf{u})^\top \mathbf{J}'(\mathbf{v}) = \tilde{\mathbf{J}}(\mathbf{u})^\top \tilde{\mathbf{J}}'(\mathbf{v}) \tag{34}$$

where all Jacobian matrices have full rank due to Prop. 2. We can hence derive

$$\mathbf{J}'(\mathbf{v}) = (\mathbf{J}(\mathbf{u})^{-\top}\tilde{\mathbf{J}}(\mathbf{u})^{\top})\tilde{\mathbf{J}}'(\mathbf{v}) \quad \mathbf{J}(\mathbf{u})^{\top} = \tilde{\mathbf{J}}(\mathbf{u})^{\top}(\tilde{\mathbf{J}}'(\mathbf{v})\mathbf{J}'(\mathbf{v})^{-1}) \tag{35}$$

$$\mathbf{J}'(\mathbf{v}) = \mathbf{A}(\mathbf{u})\tilde{\mathbf{J}}'(\mathbf{v}) \quad \mathbf{J}(\mathbf{u})^{\top} = \tilde{\mathbf{J}}(\mathbf{u})^{\top}\mathbf{B}(\mathbf{v}) \tag{36}$$

for some full rank matrices $\mathbf{A}(\mathbf{u})$ and $\mathbf{B}(\mathbf{v})$. Because the left hand side do not depend on the argument of $\mathbf{A}$ and $\mathbf{B}$, both matrices need to be constant, leaving

$$\mathbf{J}'(\mathbf{v}) = \mathbf{A}\tilde{\mathbf{J}}'(\mathbf{v}) \quad \mathbf{J}(\mathbf{u})^{\top} = \tilde{\mathbf{J}}(\mathbf{u})^{\top}\mathbf{B} \tag{37}$$

from which it follows that

$$\mathbf{h}(\mathbf{u}) = \mathbf{B}^{-1}\tilde{\mathbf{h}}(\mathbf{u}) \quad \mathbf{h}'(\mathbf{v}) = \mathbf{A}\tilde{\mathbf{h}}'(\mathbf{v}). \tag{38}$$

concluding the proof. □

Note that the previous proposition applies even when the data generating functions (i.e., the datasets) between the model run differ, and merely the latent distributions match. In this case, the same latent $\mathbf{u}_0$ can be mapped to different points $\mathbf{x}_0$ and $\tilde{\mathbf{x}}_0$ and the resulting embedding points would still satisfy $\mathbf{f}(\mathbf{u}_0) = \mathbf{A}\tilde{\mathbf{f}}(\mathbf{u}_0)$. For this reason, the proposition is written w.r.t. the composition $\tilde{\mathbf{h}}$ of data generating process and encoder. If the data generating processes match, it is clear that Eq. 38 can be equivalently written as $\mathbf{f}(\mathbf{x}) = \mathbf{B}^{-1}\tilde{\mathbf{f}}(\mathbf{x})$, $\mathbf{f}'(\mathbf{y}) = \mathbf{A}\tilde{\mathbf{f}}'(\mathbf{y})$, which matches the statement by Roeder et al. (13) (but for our modified diversity condition).

For symmetric encoders, we can give the following result:

Proposition 5. *Assume that the encoders $\mathbf{f} = \mathbf{f}', \tilde{\mathbf{f}} = \tilde{\mathbf{f}}'$ are shared, and $-\phi$ is a norm, $\phi(\mathbf{a}, \mathbf{b}) = -\|\mathbf{a} - \mathbf{b}\|$. Assume that the model minimizes the InfoNCE loss and assume that the co-domain of $\mathbf{f}$, $\mathbf{f}'$ is a normed space over $\mathbb{R}^d$. Then $\mathbf{h} = \mathbf{L}\tilde{\mathbf{h}}$.*

Proof. We write the data in terms of the underlying latents $\mathbf{x} = \mathbf{g}(\mathbf{u})$ and $\mathbf{y} = \mathbf{g}'(\mathbf{v})$. At the minimizer of the InfoNCE loss it then holds that

$$\|\mathbf{h}(\mathbf{u}) - \mathbf{h}(\mathbf{v})\| + C(\mathbf{u}) = \|\tilde{\mathbf{h}}(\mathbf{u}) - \tilde{\mathbf{h}}(\mathbf{v})\| + \tilde{C}(\mathbf{u}). \tag{39}$$

for all points in the dataset. Inserting $\mathbf{v} = \mathbf{u}$ gives $C(\mathbf{u}) = \tilde{C}(\mathbf{u})$. Because $\mathbf{h}, \tilde{\mathbf{h}}$ bijective, we define points $\mathbf{a} = \tilde{\mathbf{h}}(\mathbf{u})$ and $\mathbf{b} = \tilde{\mathbf{h}}(\mathbf{v})$, and it holds

$$\|\mathbf{h}(\tilde{\mathbf{h}}^{-1}(\mathbf{a})) - \mathbf{h}(\tilde{\mathbf{h}}^{-1}(\mathbf{b}))\| = \|\mathbf{a} - \mathbf{b}\|. \tag{40}$$

Due to the Mazur–Ulam theorem, the map $\mathbf{h} \circ \tilde{\mathbf{h}}^{-1}$ is then affine, concluding the proof. □

Note that all results in this section are also independent of the mixing functions $\mathbf{g}$ and $\mathbf{g}'$. This means that *consistency can be guaranteed irrespective of the exact data modality and generative process*, as long as the *underlying* latent distribution matches. This is given in well-controlled experiments (as one assumes when e.g., repeating experiments).

CEBRA models recover the ground-truth latents. In contrast to the identifiability results in the previous section (which made a connection between two models trained on the same or similar data distribution), in this section we will make a connection between the *ground truth model* and the model trained on the data. To make this connection, assumptions are needed about the ground truth model.

As introduced in Def. 1, we will denote the ground truth model(s) as $\mathbf{g}$ and $\mathbf{g}'$, and data from these models is then encoded with $\mathbf{f}$ and $\mathbf{f}'$, respectively. Our goal is to understand properties of their composition $\mathbf{h} = \mathbf{g} \circ \mathbf{f}$ and $\mathbf{h}' = \mathbf{g}' \circ \mathbf{f}'$, and when this composition reduces to an affine or linear transformation. Based on the property of the model (especially the similarity measure), other guarantees are possible. Towards the end of this subsection we will discuss CEBRA model setups where we obtain guarantees up to permutations and sign-flips and point-wise non-linear transformations by leveraging existing contrastive learning theory (14, 25). We base our theory on the identifiability proofs of contrastive learning given by Hyvärinen et al. (14) and Zimmermann et al. (25), and give new proofs to complete the theory for the most important usage modes in CEBRA. Note that the theory also extends to settings not explicitly demonstrated in this paper, but integrated into the CEBRA software package (e.g., trainable similarity functions ϕ).

One key property of CEBRA is its distinction of discovery-driven and hypothesis-driven training (or hybrid training, which is a combination of both where the feature encoders need to minimize both the time-contrastive and behavior-contrastive objectives). For time- and behavior-contrastive learning, similar theory applies. A key difference is that in time-contrastive (discovery-driven) learning, an underlying distribution of the latents is inherent to the dataset and "given" by the temporal

variation in the data. If it is desirable to recover the ground truth latents, the similarity measure needs to be suitable to allow full InfoNCE minimization, e.g., by model selection on basis of the InfoNCE or goodness of fit metric. Note that a cosine similarity measure is already quite flexible for a wide range of distributions and a good default, and also note that InfoNCE minimization is known to be empirically robust to minor violations between the model assumptions and ground-truth conditional distribution (25). For hypothesis-driven learning, CEBRA will always be able to find a suitable learning setup that recovers the auxiliary variables: The positive and negative sample distributions can be chosen and selected such that the propositions in this section will hold. Even if empirical distributions are used (e.g., the "time delta" distribution outlined in the Methods), it is possible to check that this distribution matches the similarity measure of the model.

Because we make statements about the relationship between continual and discrete context variables ($\mathbf{c}_t$ and k_t) and the signal space ($\mathbf{s}_t$) for each time-point t, we will again use this notation from the Methods; we use the variable names introduced in Def. 1 to denote the underlying ground-truth latents, and the samples fed to the model (which can, but do not necessarily need to, equal to the signal). We start our discussion with discovery-driven training of CEBRA using time information:

Proposition 6 (Discovery-driven CEBRA)**.** *Assume the learning setup in Def. 1, and that the ground-truth latents $\mathbf{u}_1, \ldots, \mathbf{u}_T$ for each time point follow a uniform marginal distribution and the change between subsequent time steps is given by the conditional distribution of the form*

$$p(\mathbf{u}_{t+\Delta t}|\mathbf{u}_t) = \frac{1}{Z(\mathbf{u}_t)} \exp \delta(\mathbf{u}_{t+\Delta t}, \mathbf{u}_t) \tag{41}$$

where δ is either a (scaled) dot product (and $\mathbf{u}_t \in \mathcal{S}^{n-1} \subset \mathbb{R}^d$ lies on the $(n-1)$-sphere $\mathcal{S}^{n-1}$) or an arbitrary semi-metric (and $\mathbf{u}_t \in \mathcal{U} \subset \mathbb{R}^d$ lies in a convex body $\mathcal{U}$). Assume that the data generating process $\mathbf{g}$ with $\mathbf{s}_t = \mathbf{g}(\mathbf{u}_t)$ is injective. Assume we train a symmetric CEBRA model with encoder $\mathbf{f} = \mathbf{f}'$ and the similarity measure including a fixed temperature $\tau > 0$ is set to or sufficiently flexible such that $\phi = \delta$ for all arguments. Then $\mathbf{h} = \mathbf{h}' = \mathbf{g} \circ \mathbf{f}$ is affine.

Proof. For δ being the dot product, the result follows from the proof of Theorem 2 in Zimmermann et al. (25). For δ being a semi-metric, the result follows from the proof of Theorem 5 in Zimmermann et al. (25). □

We will next consider the hypothesis-driven mode in CEBRA. Here, we either choose a parametric or non-parametric positive distribution p to shape the embedding space. Our goal is find an embedding space reflecting the auxiliary variable, in case there is a meaningful relationship between the signal and this variable.

Naturally, the actual signal will depend on additional latents that we do not record as auxiliary information. The full data generating process can be written as

$$\mathbf{s}_t = \mathbf{g}(\mathbf{c}_t, k_t, \mathbf{z}_t) \tag{42}$$

where $\mathbf{z}_t$ are additional latent sources not observed during training. Since $\mathbf{g}$ is an injective function, it follows that $\mathbf{s}_t = \mathbf{s}_{t'}$ implies $\mathbf{c}_t = \mathbf{c}_{t'}$, $k_t = k_{t'}$, $\mathbf{z}_t = \mathbf{z}_{t'}$. Applying hypothesis-driven learning in this setup will force an embedding space representing $\mathbf{c}$ and k, but not $\mathbf{z}$. We can denote the set $G(\mathbf{c}, k) := \{\mathbf{z} \in Z : \mathbf{g}(\mathbf{c}, k, \mathbf{z})\}$ which contains all points in signal space corresponding to a particular set of auxiliary variables $\mathbf{c}, k$. Because $\mathbf{g}$ is injective, there exists a function $\tilde{\mathbf{g}}$, which will retrieve the original auxiliary variables: $\tilde{\mathbf{g}}(\mathbf{g}(\mathbf{c}, k, \mathbf{z})) = (\mathbf{c}, k)^\top$.

Proposition 7 (Hypothesis-driven CEBRA)**.** *Assume a partially observable data generating process, where*

$$\mathbf{y} = \mathbf{g}'(\mathbf{c}, \mathbf{v}) \tag{43}$$

with $\mathbf{g}'$ injective and with $\mathbf{c}$ as an observable context variable, and $\mathbf{v}$ as a latent. As in Prop. 6 assume that $\mathbf{c}$ lies on a hypersphere if ϕ is the dot-product similarity, and on a convex body if $-\phi$ is a semi-metric. Then the minimizer of the InfoNCE loss trained with a distribution $p(\mathbf{c}|\tilde{\mathbf{c}}) = \exp(\phi(\mathbf{c}, \tilde{\mathbf{c}}))/Z(\tilde{\mathbf{c}})$ and a uniform marginal and negative distribution $q = p_{\mathcal{D}}$ will yield a $\mathbf{h}$ that recovers $\mathbf{c}$ up to an affine transformation.

Proof. $\mathbf{h}$ is the composition $\mathbf{g} \circ \mathbf{f}$. Per assumption, the similarity function ϕ is sufficiently flexible such that

$$\phi(\mathbf{h}(\mathbf{c}, \mathbf{z}), \mathbf{h}(\tilde{\mathbf{c}}, \mathbf{z})) = \log p(\mathbf{c}, \tilde{\mathbf{c}}) + C(\tilde{\mathbf{c}}) \quad \forall \mathbf{z}. \tag{44}$$

Because $\mathbf{g}$ is injective, $\mathbf{g}(\mathbf{c}, \mathbf{z}) = \mathbf{g}(\mathbf{c}', \mathbf{z}')$ implies that $\mathbf{c} = \mathbf{c}'$ and $\mathbf{z} = \mathbf{z}'$ and there exists a function $\tilde{\mathbf{g}}$ such that $\tilde{\mathbf{g}}(\mathbf{g}(\mathbf{c}, \mathbf{z})) = \mathbf{c}$ for all $\mathbf{z}$. It remains to show that $\mathbf{f} = \mathbf{L}\tilde{\mathbf{g}}$ at the minimizer for a full-rank matrix $\mathbf{L}$. By comparing arguments on the left hand side and right and side, and inserting the form of p, solutions of Eq. 44 need to simplify to

$$\phi(\mathbf{h}(\mathbf{c}), \mathbf{h}(\tilde{\mathbf{c}})) = \phi(\mathbf{c}, \tilde{\mathbf{c}}) + C(\tilde{\mathbf{c}}) \tag{45}$$

By symmetry of ϕ, $C(\tilde{\mathbf{c}}) = 0$ vanishes. Then the result follows again from Theorem 2 in Zimmermann et al. (25) for ϕ being the dot-product similarity, and from Theorem 5 in Zimmermann et al. (25) for $-\phi$ being a semi-metric. □

With our results from Prop. 1 and Prop. 2, it is also possible for us to extend the results for distributions fulfilling the diversity condition for bijectivity (Def. 4). The following proposition enables a statement about distribution where the data generating process and/or feature encoder differs between the reference and positive/negative pairs. An example used in the main text is multi-session training, or training across data modalities. We first need to strengthen the assumption about the positive and negative distributions from Def. 4. Again assume that the latent dimensions $d = d'$ match.

Definition 5 (Strong assumption of diversity for bijectivity)**.** *The sampling process with distributions p and q satisfies*

$$\left[\frac{\partial^2 \log p(\mathbf{v}|\mathbf{u})}{\partial u_i \partial v_j}\right]_{i\in[d],j\in[d]} - \left[\frac{\partial^2 \log q(\mathbf{v}|\mathbf{u})}{\partial u_i \partial v_j}\right]_{i\in[d],j\in[d]} = \mathbf{L} \tag{46}$$

for some full-rank matrix $\mathbf{L} \in \mathbb{R}^{d\times d}$ for all $\mathbf{u} \in \mathbb{R}^d$ in the support of the marginal distribution $p_{\mathcal{D}}$.

Note that Def. 5 is a special case of the diversity condition for bijectivity in Def. 4. The stronger condition still covers important distributions like a Gaussian conditional, even with an additional mean and covariance $p(\mathbf{v}|\mathbf{u}) = \exp(-(\mathbf{u}-\mathbf{v}-\mu)^\top \mathbf{\Sigma}^{-1}(\mathbf{u}-\mathbf{v}-\mu))$. Likewise, it would cover von Mises-Fisher (vMF) distributions, even with a rotated reference sample and $p(\mathbf{v}|\mathbf{u}) = \exp(\kappa \mathbf{u}^\top \mathbf{Q}\mathbf{v})$. We will cover these selected special cases using the stronger diversity assumption below.

Proposition 8. *Consider a data generating process with uniform marginal and positive/negative distributions satisfying Def. 5. Assume the data is generated by mixing functions $\mathbf{g} : \mathbb{R}^d \mapsto \mathbb{R}^D$ and $\mathbf{g}' \in \mathbb{R}^d \mapsto \mathbb{R}^{D'}$ and encoded into a shared E-dimensional embedding space with two separate encoders $\mathbf{f} : \mathbb{R}^D \mapsto \mathbb{R}^E$ and $\mathbf{f}' : \mathbb{R}^{D'} \mapsto \mathbb{R}^E$ and assume that ϕ is the dot-product. Then there are d dimensions in $\mathbf{h}$, and d dimensions in $\mathbf{h}'$ which represent the latents up to an affine transform.*

Proof. Let us again denote the compositions $\mathbf{h} = \mathbf{g} \circ \mathbf{f} : \mathbb{R}^d \mapsto \mathbb{R}^E$ and $\mathbf{h}' = \mathbf{g}' \circ \mathbf{f}' : \mathbb{R}^d \mapsto \mathbb{R}^E$ and let us denote the Jacobian matrices as $\mathbf{J} : \mathbb{R}^d \mapsto \mathbb{R}^{E\times d}$ and $\mathbf{J}' : \mathbb{R}^d \mapsto \mathbb{R}^{E\times d}$, respectively. Without loss of generality (the indices can be arbitrarily permuted), let us split each $\mathbf{h}$ into two parts, with $\mathbf{h}_1 := [h_1, \dots, h_d]^\top$ and $\mathbf{h}_2 := [h_{d+1}, \dots, h_E]^\top$ (in case $E > d$), and respectively for $\mathbf{h}'$. At the minimizer of the InfoNCE loss, we get the condition

$$\mathbf{h}(\mathbf{u})^\top \mathbf{h}'(\mathbf{v})/\tau = \log \frac{p(\mathbf{v}|\mathbf{u})}{q(\mathbf{v}|\mathbf{u})} + C(\mathbf{u}). \tag{47}$$

We take the derivative w.r.t. $\mathbf{v}$ on both sides, with gives

$$\mathbf{J}'(\mathbf{v})^\top \mathbf{h}(\mathbf{u})/\tau = \frac{\partial}{\partial \mathbf{v}} \left[\log \frac{p(\mathbf{v}|\mathbf{u})}{q(\mathbf{v}|\mathbf{u})}\right], \tag{48}$$

with $\mathbf{J}'$ denoting the Jacobian matrix of $\mathbf{h}'$. We now take the derivative w.r.t. $\mathbf{u}$ on both sides, with gives

$$\mathbf{J}(\mathbf{u})^\top \mathbf{J}'(\mathbf{v})/\tau = \frac{\partial^2}{\partial \mathbf{u} \partial \mathbf{v}} \left[\log \frac{p(\mathbf{v}|\mathbf{u})}{q(\mathbf{v}|\mathbf{u})}\right]. \tag{49}$$

and by assumption we can insert Def. 5 and re-arrange to

$$\mathbf{J}(\mathbf{u})\mathbf{J}'(\mathbf{v})^\top = \tau \mathbf{L} \tag{50}$$

for some full-rank matrix $\mathbf{L} \in \mathbb{R}^{d\times d}$. Note that when $E > d$ this condition is underconstrained. Without loss of generality, let us split the Jacobian matrices into two parts:

$$\mathbf{J}(\mathbf{u}) := [\mathbf{J}_1(\mathbf{u}); \mathbf{0}] \quad \mathbf{J}'(\mathbf{v}) := [\mathbf{J}'_1(\mathbf{v}); \mathbf{J}'_2(\mathbf{v})] \tag{51}$$

where $\mathbf{J}_1 : \mathbb{R}^d \mapsto \mathbb{R}^{d\times d}$, $\mathbf{J}'_1 : \mathbb{R}^d \mapsto \mathbb{R}^{d\times d}$, $\mathbf{J}'_2 : \mathbb{R}^d \mapsto \mathbb{R}^{(E-d)\times d}$. This allows to re-write the condition as

$$\mathbf{J}_1(\mathbf{u})\mathbf{J}'_1(\mathbf{v})^\top = \tau \mathbf{L} \tag{52}$$

which is no longer underconstrained. This equation is valid for any $\mathbf{u}, \mathbf{v}$ in the support of the marginal and the positive/negative distribution, which is why the left hand side cannot depend on $\mathbf{u}$ and $\mathbf{v}$, leaving the final form of the Jacobians:

$$\mathbf{J}(\mathbf{u}) := [\mathbf{J}_1; \mathbf{0}] \quad \mathbf{J}'(\mathbf{v}) := [\mathbf{J}'_1; \mathbf{a}(\mathbf{v})], \tag{53}$$

where $\mathbf{a} : \mathbb{R}^d \mapsto \mathbb{R}^{(E-d)\times d}$ is an arbitrary function that ensures that Eq. 47 can hold. It follows that $\mathbf{h}_1$ and $\mathbf{h}'_1$ are affine transforms, $\mathbf{h}_2$ is a constant, and $\mathbf{h}'_2$ will be a potentially non-linear transform to match the InfoNCE minimizer, concluding the proof. □

We next discuss two special cases that are of interest to users of CEBRA and demonstrate that the previous result is more flexible than the results presented in Zimmermann et al. (25). Consider a setup where we have a constant "offset" between the latents vs. having a perfectly symmetric p. We are still able to recover the underlying latents for both vMF and Normal conditional distributions:

Corollary 1. *The aforementioned result holds for vMF distributions with a constant offset between* $\mathbf{u}$ *and* $\mathbf{v}$ *on the hypersphere, parameterized by a rotation matrix* $\mathbf{Q}$*, with* $\log p(\mathbf{v}|\mathbf{u}) = \kappa \mathbf{u}^\top \mathbf{Q}\mathbf{v}$*. Assume* $d = d' = k$ *and assume the domain and co-domain of* $\mathbf{h}$*,* $\mathbf{h}'$ *is the unit sphere.*

Proof. The distributions satisfies constancy of the second derivative, and the condition on the Jacobian matrices is

$$\mathbf{J}_1 \mathbf{J}_1'^\top = (\tau\kappa)\mathbf{Q} \tag{54}$$

and since all vectors $\mathbf{h}(\mathbf{a}) = \mathbf{J}^\top \mathbf{a}$ need to be normalized and $\mathbf{Q}$ is orthogonal, we can consider any column $\mathbf{q}_i$ and the corresponding vector $\mathbf{a}_i$ of $\mathbf{J}_1$

$$\mathbf{J}_1 \mathbf{a}_i = (\tau\kappa)\mathbf{q}_i \Rightarrow \|\mathbf{J}_1 \mathbf{a}_i\|^2 = (\tau\kappa)\|\mathbf{q}_i\|^2 \Rightarrow 1 = (\tau\kappa)1 \tag{55}$$

and it follows $\tau = 1/\kappa$. □

Corollary 2. *The aforementioned result holds for a Gaussian distribution with a constant offset between* $\mathbf{u}$ *and* $\mathbf{v}$ *parameterized, with* $\log p(\mathbf{v}|\mathbf{u}) = \|\mathbf{u} - \mathbf{v} - \mu\|^2$*. Assume* $d = d' = k$*.*

Proof. We rewrite the log-conditional as

$$\begin{aligned}
\log p(\mathbf{v}|\mathbf{u}) &= -\|(\mathbf{u} - \mathbf{v}) - \boldsymbol{\mu}\|^2 && (56)\\
&= -\|\mathbf{u} - \mathbf{v}\|^2 - \|\boldsymbol{\mu}\|^2 + 2(\mathbf{u} - \mathbf{v})^\top \boldsymbol{\mu} && (57)\\
&= -\|\mathbf{u}\|^2 - \|\mathbf{v}\|^2 - \|\boldsymbol{\mu}\|^2 + 2\mathbf{u}^\top \boldsymbol{\mu} + 2\mathbf{v}^\top \boldsymbol{\mu} + 2\mathbf{u}^\top \mathbf{v} && (58)\\
& && (59)
\end{aligned}$$

which gives, at the minimizer of the InfoNCE objective,

$$\mathbf{h}(\mathbf{u})^\top \mathbf{h}'(\mathbf{v})/\tau = 2\mathbf{u}^\top \mathbf{v} + (-\|\mathbf{v}\|^2 + 2\mathbf{v}^\top \boldsymbol{\mu}) + (-\|\mathbf{u}\|^2 + 2\mathbf{u}^\top \boldsymbol{\mu} + C(\mathbf{u})) \tag{60}$$

and we recover

$$\begin{aligned}
\mathbf{h}_1(\mathbf{u}) &= \mathbf{J}_1 \mathbf{u}, && (61)\\
\mathbf{h}_1'(\mathbf{v}) &= \mathbf{J}_1' \mathbf{v}, && (62)\\
C(\mathbf{u}) &= \|\mathbf{u}\|^2 - 2\mathbf{u}^\top \boldsymbol{\mu} && (63)\\
\mathbf{h}_2'(\mathbf{v}) &= -\|\mathbf{v}\|^2 + 2\mathbf{v}^\top \boldsymbol{\mu}. && (64)
\end{aligned}$$

□

While the above results extended the theory of Zimmermann et al. (25) to training setups within the CEBRA library and used in the main text (training with dot-product and negative mean squared error as the similarity measure and sampling procedure), we can also leverage previous results from Hyvärinen et al. (14) to further extend the families of possible distributions.

Firstly, for conditional exponential family distributions within an ICA framework, the dot-product similarity can be used in conjunction with non-symmetric encoders $\mathbf{f}$, $\mathbf{f}'$ (i.e., two separate networks) to recover the components up to a linear indeterminacy. We recall Definition 1 from Hyvärinen et al. (14):

Definition C (Conditionally exponential distributions (Def. 1 from Hyvärinen et al. (14)))**.** *A random variable (independent component)* v_i *is conditionally exponential of order* k *given random vector* $\mathbf{x}$ *if its conditional probability density function can be given in the form*

$$p(v_i|\mathbf{x}) = \frac{Q_i(v_i)}{Z_i(\mathbf{x})} \exp\left[\sum_{j=1}^{k} \tilde{q}_{ij}(v_i)\lambda_{ij}(\mathbf{x})\right] \tag{65}$$

almost everywhere in the support of $\mathbf{x}$*, with* $\tilde{q}_{ij}$*,* λ_{ij}*,* Q_i*, and* Z_i *scalar-valued functions. The sufficient statistics* $\tilde{q}_{ij}$ *are assumed linearly independent (over* j*, for each fixed* i*).*

We observe a similar functional form as used in Prop. 8. Values of $\tilde{q}_{ij}(v_i)$ would be represented by $\mathbf{f}'$, and values in $\lambda_{ij}(\mathbf{x})$ would be represented by $\mathbf{f}$. As in the previous proposition, more dimensions in $\mathbf{f}$, $\mathbf{f}'$ as latent variables are required ($E > d$) for representing conditional distributions of the conditional exponential form. In this setup, Theorem 3 from Hyvärinen et al. (14) implies in our context that the sufficient statistics of the latents can be recovered up to a linear transformation.

Secondly, for arbitrary conditional distributions, as long as variability conditions are satisfied within an ICA framework, contrastive learning can recover the underlying components up to a permutation and point-wise invertible transformation (14). Using this mode applies when a similarity measure ϕ which gives

$$\psi(\mathbf{u},\mathbf{v}) := \sum_{i=1}^{E} \phi_i(h_i'(\mathbf{v}), \mathbf{h}(\mathbf{u})), \text{ or w.r.t. to the signal variables, } \psi(\mathbf{x},\mathbf{y}) = \sum_{i=1}^{E} \phi_i(f_i'(\mathbf{y}), \mathbf{f}(\mathbf{x})), \tag{66}$$

is used within the CEBRA framework. In this case, Theorem 1 in Hyvärinen et al. (14) applies and gives an identifiability guarantee of $\mathbf{h}$ to recover the ground truth latents $\mathbf{v}$ up to permutations and component-wise non-linear transformations.

Supplementary Tables

Supplementary Table 1. Consistency statistics related to Fig. 1. Data includes all rats (n=4).

group 1	group 2	P-value	Reject
CEBRA-Behavior	CEBRA-Time	0.0124	**True**
CEBRA-Behavior	conv-pi-VAE w/labels	0.0128	**True**
CEBRA-Behavior	conv-pi-VAE without	1.4e-10	**True**
CEBRA-Behavior	tSNE	2.2e-05	**True**
CEBRA-Behavior	UMAP	2.1e-14	**True**
CEBRA-Time	conv-pi-VAE w/labels	.99	False
CEBRA-Time	conv-pi-VAE without	0.0001	**True**
CEBRA-Time	tSNE	0.45	False
CEBRA-Time	UMAP	1.02e-08	**True**

Supplementary Table 2. Decoding Statistics related to Fig. 2. Data includes all rats (n=4); supervised grouping one way ANOVA F-55, p=4.7e-31; self- and unsupervised, one way ANOVA F=8, p=6.95e-05. Posthoc Tukey HSD Tests:

group 1	group 2	P-value	Reject
CEBRA-Behavior	conv-pi-VAE (MC decoding)	0.9	False
CEBRA-Behavior	conv-pi-VAE (kNN)	0.001	**True**
CEBRA-Behavior	pi-VAE (MC decoding)	0.001	**True**
CEBRA-Behavior	pi-VAE (kNN)	0.001	**True**
CEBRA-Time	PCA	0.0024	**True**
CEBRA-Time	tSNE	0.0021	**True**
CEBRA-Time	UMAP	0.0057	**True**

Supplementary Table 3. Related to Fig. 5. Posthoc Tukey HSD Test Allen Neuropixels dataset, 10 Frame window (< 30 neurons=False):

neuron no.	group 1	group 2	P-value	Reject
30	baseline-bayes	baseline-knn	0.016	**True**
30	baseline-bayes	CEBRA	0.1255	False
30	baseline-knn	CEBRA	0.7083	False
30	baseline-bayes	CEBRA-joint	0.0072	**True**
30	baseline-knn	CEBRA-joint	0.9784	False
30	CEBRA	CEBRA-joint	0.4762	False
50	baseline-bayes	baseline-knn	0.0358	**True**
50	baseline-bayes	CEBRA	0.324	False
50	baseline-knn	CEBRA	0.5956	False
50	baseline-bayes	CEBRA-joint	0.1296	False
50	baseline-knn	CEBRA-joint	0.8989	False
50	CEBRA	CEBRA-joint	0.9379	False
100	baseline-bayes	baseline-knn	0.0	**True**
100	baseline-bayes	CEBRA	0.2589	False
100	baseline-knn	CEBRA	0.0002	**True**
100	baseline-bayes	CEBRA-joint	0.372	False
100	baseline-knn	CEBRA-joint	0.0001	**True**
100	CEBRA	CEBRA-joint	0.9941	False
200	baseline-bayes	baseline-knn	0.0	**True**
200	baseline-bayes	CEBRA	0.9976	False
200	baseline-knn	CEBRA	0.0	**True**
200	baseline-bayes	CEBRA-joint	0.7999	False
200	baseline-knn	CEBRA-joint	0.0	**True**
200	CEBRA	CEBRA-joint	0.6964	False
400	baseline-bayes	baseline-knn	0.0004	**True**
400	baseline-bayes	CEBRA	0.2531	False
400	baseline-knn	CEBRA	0.0	**True**
400	baseline-bayes	CEBRA-joint	0.2166	False
400	baseline-knn	CEBRA-joint	0.0	**True**
400	CEBRA	CEBRA-joint	0.9996	False
600	baseline-bayes	baseline-knn	0.0002	**True**
600	baseline-bayes	CEBRA	0.2095	False
600	baseline-knn	CEBRA	0.0	**True**
600	baseline-bayes	CEBRA-joint	0.2884	False
600	baseline-knn	CEBRA-joint	0.0	**True**
600	CEBRA	CEBRA-joint	0.9967	False
800	baseline-bayes	baseline-knn	0.0001	**True**
800	baseline-bayes	CEBRA	0.3691	False
800	baseline-knn	CEBRA	0.0	**True**
800	baseline-bayes	CEBRA-joint	0.1668	False
800	baseline-knn	CEBRA-joint	0.0	**True**
800	CEBRA	CEBRA-joint	0.9524	False
900	baseline-bayes	baseline-knn	0.0003	**True**
900	baseline-bayes	CEBRA	0.3867	False
900	baseline-knn	CEBRA	0.0	**True**
900	baseline-bayes	CEBRA-joint	0.3522	False
900	baseline-knn	CEBRA-joint	0.0	**True**
900	CEBRA	CEBRA-joint	0.9999	False
1000	baseline-bayes	baseline-knn	0.0018	**True**
1000	baseline-bayes	CEBRA	0.4707	False
1000	baseline-knn	CEBRA	0.0001	**True**
1000	baseline-bayes	CEBRA-joint	0.3785	False
1000	baseline-knn	CEBRA-joint	0.0001	**True**
1000	CEBRA	CEBRA-joint	0.9981	False

Supplementary Table 4. Related to Fig. 5. Posthoc Tukey HSD Test Allen Neuropixels dataset, 1 Frame window (below 50 neurons all False):

neuron no.	group 1	group 2	P-value	Reject
50	baseline-bayes	baseline-knn	0.0233	**True**
50	baseline-bayes	CEBRA	0.8918	False
50	baseline-knn	CEBRA	0.0055	True
50	baseline-bayes	CEBRA-joint	0.8238	False
50	baseline-knn	CEBRA-joint	0.004	**True**
50	CEBRA	CEBRA-joint	0.9987	False
100	baseline-bayes	baseline-knn	0.0	**True**
100	baseline-bayes	CEBRA	0.0275	**True**
100	baseline-knn	CEBRA	0.0132	**True**
100	baseline-bayes	CEBRA-joint	0.9977	False
100	baseline-knn	CEBRA-joint	0.0	**True**
100	CEBRA	CEBRA-joint	0.0395	**True**
200	baseline-bayes	baseline-knn	0.0005	**True**
200	baseline-bayes	CEBRA	0.3703	False
200	baseline-knn	CEBRA	0.0	**True**
200	baseline-bayes	CEBRA-joint	0.0058	**True**
200	baseline-knn	CEBRA-joint	0.0	**True**
200	CEBRA	CEBRA-joint	0.1478	False
400	baseline-bayes	baseline-knn	0.0044	**True**
400	baseline-bayes	CEBRA	0.0336	**True**
400	baseline-knn	CEBRA	0.0	**True**
400	baseline-bayes	CEBRA-joint	0.0	**True**
400	baseline-knn	CEBRA-joint	0.0	**True**
400	CEBRA	CEBRA-joint	0.0	**True**
600	baseline-bayes	baseline-knn	0.0125	**True**
600	baseline-bayes	CEBRA	0.6063	False
600	baseline-knn	CEBRA	0.001	**True**
600	baseline-bayes	CEBRA-joint	0.0	**True**
600	baseline-knn	CEBRA-joint	0.0	**True**
600	CEBRA	CEBRA-joint	0.0	**True**
800	baseline-bayes	baseline-knn	0.0006	**True**
800	baseline-bayes	CEBRA	0.0008	**True**
800	baseline-knn	CEBRA	0.0	**True**
800	baseline-bayes	CEBRA-joint	0.0	**True**
800	baseline-knn	CEBRA-joint	0.0	**True**
800	CEBRA	CEBRA-joint	0	**True**
900	baseline-bayes	baseline-knn	0.0004	**True**
900	baseline-bayes	CEBRA	0.0048	**True**
900	baseline-knn	CEBRA	0.0	**True**
900	baseline-bayes	CEBRA-joint	0.0	**True**
900	baseline-knn	CEBRA-joint	0.0	**True**
900	CEBRA	CEBRA-joint	0.0	**True**
1000	baseline-bayes	baseline-knn	0.0	**True**
1000	baseline-bayes	CEBRA	0.0	**True**
1000	baseline-knn	CEBRA	0.0	**True**
1000	baseline-bayes	CEBRA-joint	0.0	**True**
1000	baseline-knn	CEBRA-joint	0.0	**True**
1000	CEBRA	CEBRA-joint	0.0019	**True**

Supplementary Table 5. Related to Fig. 5. Posthoc Tukey HSD Test Allen Neuropixels dataset, scene classification with 1 Frame window:

neuron no.	group 1	group 2	P-value	Reject
50	baseline-bayes	baseline-knn	0.4575	False
50	baseline-bayes	CEBRA	0.1986	False
50	baseline-knn	CEBRA	0.9355	False
50	baseline-bayes	CEBRA-joint	0.0	**True**
50	baseline-knn	CEBRA-joint	0.0	**True**
50	CEBRA	CEBRA-joint	0.0	**True**
100	baseline-bayes	baseline-knn	0.0207	**True**
100	baseline-bayes	CEBRA	0.0084	**True**
100	baseline-knn	CEBRA	0.9694	False
100	baseline-bayes	CEBRA-joint	0.0	**True**
100	baseline-knn	CEBRA-joint	0.0	**True**
100	CEBRA	CEBRA-joint	0.0	**True**
200	baseline-bayes	baseline-knn	0.0106	**True**
200	baseline-bayes	CEBRA	0.0001	**True**
200	baseline-knn	CEBRA	0.1269	False
200	baseline-bayes	CEBRA-joint	0.0	**True**
200	baseline-knn	CEBRA-joint	0.0	**True**
200	CEBRA	CEBRA-joint	0.0	**True**
400	baseline-bayes	baseline-knn	0.0047	**True**
400	baseline-bayes	CEBRA	0.0001	**True**
400	baseline-knn	CEBRA	0.239	False
400	baseline-bayes	CEBRA-joint	0.0	**True**
400	baseline-knn	CEBRA-joint	0.0	**True**
400	CEBRA	CEBRA-joint	0.0	**True**
600	baseline-bayes	baseline-knn	0.0013	**True**
600	baseline-bayes	CEBRA	0.0	**True**
600	baseline-knn	CEBRA	0.0032	**True**
600	baseline-bayes	CEBRA-joint	0.0	**True**
600	baseline-knn	CEBRA-joint	0.0	**True**
600	CEBRA	CEBRA-joint	0.0	**True**
800	baseline-bayes	baseline-knn	0.0	**True**
800	baseline-bayes	CEBRA	0.0	**True**
800	baseline-knn	CEBRA	0.0	**True**
800	baseline-bayes	CEBRA-joint	0.0	**True**
800	baseline-knn	CEBRA-joint	0.0	**True**
800	CEBRA	CEBRA-joint	0.0	**True**
900	baseline-bayes	baseline-knn	0.0062	**True**
900	baseline-bayes	CEBRA	0.0	**True**
900	baseline-knn	CEBRA	0.0168	**True**
900	baseline-bayes	CEBRA-joint	0.0	**True**
900	baseline-knn	CEBRA-joint	0.0	**True**
900	CEBRA	CEBRA-joint	0.0	**True**
1000	baseline-bayes	baseline-knn	0.0002	**True**
1000	baseline-bayes	CEBRA	0.0	**True**
1000	baseline-knn	CEBRA	0.0	**True**
1000	baseline-bayes	CEBRA-joint	0.0	**True**
1000	baseline-knn	CEBRA-joint	0.0	**True**
1000	CEBRA	CEBRA-joint	0.0	**True**

Supplementary Table 6. Related to Fig. 5. Posthoc Tukey HSD Test Allen Neuropixels dataset, Mean frame error, 10 frames

neuron no.	group 1	group 2	P-value	Reject
1000	baseline-bayes	baseline-knn	0.5277	False
1000	baseline-bayes	CEBRA	0.0013	**True**
1000	baseline-knn	CEBRA	0.0001	**True**
1000	baseline-bayes	CEBRA-joint	0.0011	**True**
1000	baseline-knn	CEBRA-joint	0.0001	**True**
1000	CEBRA	CEBRA-joint	0.9996	False