nature methods

ARTICLES

https://doi.org/10.1038/s41592-019-0576-7

Exploring single-cell data with deep multitasking neural networks

Matthew Amodio[1,11], David van Dijk[1,2,11], Krishnan Srinivasan[1,11], William S. Chen[3], Hussein Mohsen[4], Kevin R. Moon[5], Allison Campbell[3], Yujiao Zhao[6], Xiaomei Wang[6], Manjunatha Venkataswamy[7], Anita Desai[7], V. Ravi[7], Priti Kumar[8], Ruth Montgomery[6], Guy Wolf[9,10,11] and Smita Krishnaswamy[1,2,11]*

It is currently challenging to analyze single-cell data consisting of many cells and samples, and to address variations arising from batch effects and different sample preparations. For this purpose, we present SAUCIE, a deep neural network that combines parallelization and scalability offered by neural networks, with the deep representation of data that can be learned by them to perform many single-cell data analysis tasks. Our regularizations (penalties) render features learned in hidden layers of the neural network interpretable. On large, multi-patient datasets, SAUCIE's various hidden layers contain denoised and batch-corrected data, a low-dimensional visualization and unsupervised clustering, as well as other information that can be used to explore the data. We analyze a 180-sample dataset consisting of 11 million T cells from dengue patients in India, measured with mass cytometry. SAUCIE can batch correct and identify cluster-based signatures of acute dengue infection and create a patient manifold, stratifying immune response to dengue.

Processing single-cell data of high dimensionality and scale is inherently difficult, especially considering the degree of noise, batch effects, artifacts, sparsity and heterogeneity in the data[1,2]. Furthermore, this effect becomes exacerbated as one tries to compare between samples, which themselves contain noisy heterogeneous compositions of cellular populations. Deep learning offers promise as a technique for handling the size and dimensionality of modern biological datasets. However, deep learning has been underused for unsupervised exploratory tasks.

In this paper, we use a regularized autoencoder, which is a neural network that learns to recreate its own input via a low-dimensional bottleneck layer that learns representations of the data and enables a denoised reconstruction of the input[3–7].

Since autoencoders learn their own features, they can reveal structure in the data without defining or explicitly learning a similarity or distance metric in the original data space as other dimensionality reduction methods do (for instance, PCA uses covariance and diffusion maps[8] use affinities based on a kernel choice). We use this autoencoder approach to construct SAUCIE, a sparse autoencoder for unsupervised clustering, imputation and embedding, which is aimed to enable exploratory tasks via its design choices. SAUCIE is a multilayered deep neural network, whose input layer is fed single-cell measurements, such as mass cytometry or single-cell RNA sequencing, of an individual cell. Different layers reveal different representations of the data: for visualization, batch correction, clustering,and denoising. SAUCIE provides a unified representation of data where different aspects or features are emphasized in different layers, forming a one-step data analysis pipeline.

We apply SAUCIE to the batch correcting, denoising and clustering of an 11-million cell mass cytometry dataset with 180 samples from 40 subjects in a study of the dengue flavivirus and see the proportions of subpopulations.

Results

The SAUCIE architecture and layer regularizations. To enable unsupervised learning in a scalable manner, we base our method on the autoencoder. A key challenge is to extract meaning from the model's internal representation of the data. Specifically, we seek representations in hidden layers that are useful for performing the various analysis tasks associated with single-cell data. Here, we introduce several design decisions and novel regularizations to our autoencoder architecture (Fig. 1) to constrain the learned representations for four key tasks: clustering, batch correction, denoising and imputation, and visualization and dimensionality reduction.

For each task, dedicated design decisions are used to produce the desirable result.

Clustering. First, to cluster the data, we introduce the information dimension (ID) regularization that encourages activations of the neurons in a hidden layer of the network to be binarizable. By obtaining a 'digital' binary encoding, we can easily turn these codes into clusters. The network without regularizations tends to store its information in a distributed, or 'analog' way. With the ID regularization, the activations are all near 0 or 1, that is, binary or digital, and thus amenable to clustering by simple thresholding-based binarization (Fig. 2). This leads to a clustering of the cells that effectively represents the data space (Fig. 3). Thus, the ID regularization

[1]Department of Computer Science, Yale University, New Haven, CT, USA. [2]Department of Genetics, Yale University, New Haven, CT, USA. [3]School of Medicine, Yale University, New Haven, CT, USA. [4]Computational Biology and Bioinformatics, Yale University, New Haven, CT, USA. [5]Department of Mathematics and Statistics, Utah State University, Logan, UT, USA. [6]Department of Rheumatology, Yale University, New Haven, CT, USA. [7]Department of Neurovirology, NIMHANS, Bangalore, India. [8]Department of Microbial Pathogenesis, Yale University, New Haven, CT, USA. [9]Department of Mathematics and Statistics, Université de Montréal, Montréal, Quebec, Canada. [10]Mila – Quebec Artificial Intelligence Institute, Montréal, Quebec, Canada. [11]These authors contributed equally: Matthew Amodio, David van Dijk, Krishnan Srinivasan, Guy Wolf, Smita Krishnaswamy. *e-mail: smita.krishnaswamy@yale.edu

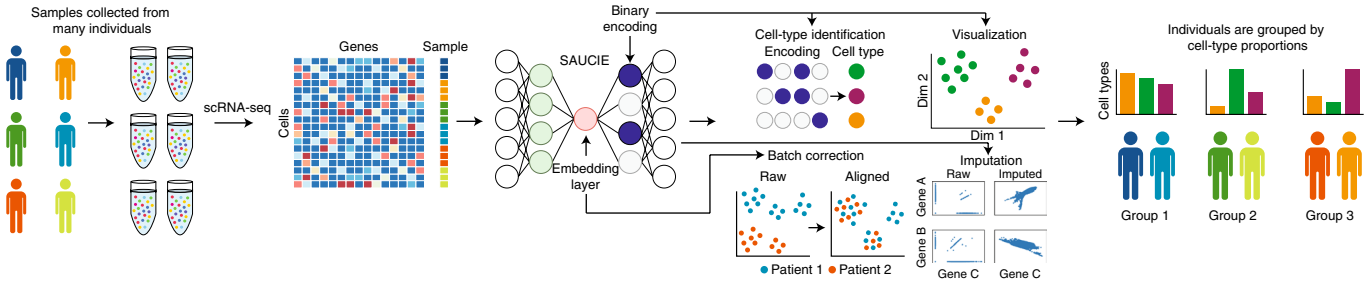

Fig. 1 | The pipeline for analyzing single-cell data in large cohorts with SAUCIE. SAUCIE performs imputation and denoising, batch effect removal, clustering and visualization on the entire cohort of patients with a unified model, and is able to provide interpretable, quantifiable metrics on each subject or group of subjects.

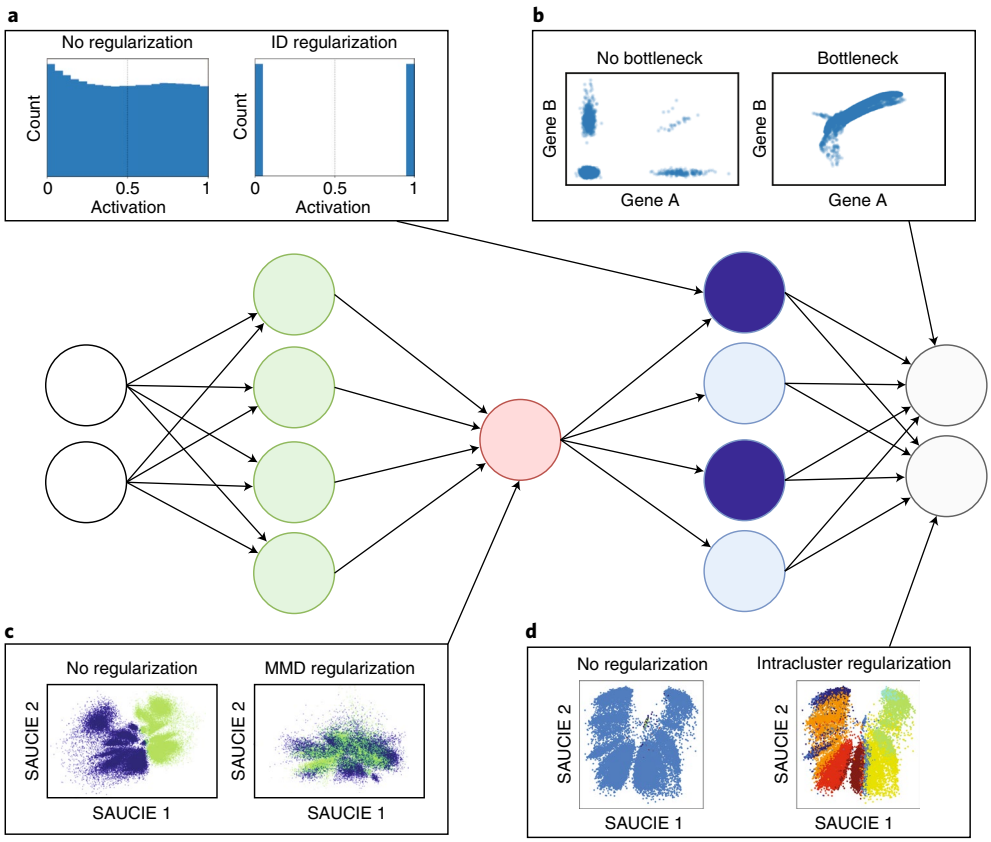

Fig. 2 | Regularizations and architecture choices in SAUCIE. a, The ID regularization applied on the sparse encoding layer produces digital codes for clustering. **b**, The informational bottleneck, that is, a smaller embedding layer, uses dimensionality reduction to produce denoised data at the output. **c**, The MMD regularization removes batch artifacts. **d**, The within-cluster distance regularization applied to the denoised data provides coherent clusters.

achieves an analog-to-digital conversion that enables interpretation of the representation as data groups or clusters corresponding to each binary code. To further encourage clusters to be homogenous groups of cells, we also introduce a within-cluster distance regularization that penalizes cells with similar clustering-layer representations being distant in the original data space. Previous work in the same vein of learning binary representations, Binary Connect, has shown promise in encouraging networks to learn in ways that are easy to binarize[9]. That work differs from SAUCIE in that they learn binary weights rather than binary activations, along with the goal to improve computational efficiency rather than achieve a clustering of the data. Further work has considered binarizing activations, as well, but do so with exact binarization (as opposed to our activations that are still continuous but are encouraged to be near binary) and do so with the aim of compressing the network into a smaller memory footprint (rather than our clustering)[10,11].

Batch correction. Batch effects are generally systematic differences found in biological data measured under different experimental runs, largely due to ambient conditions such as temperature, machine calibration or day-to-day variation in measurement efficiency. Thus, measurements even from very similar systems, such as blood cells from the same patient, appear to have a shift or difference between two different experimental runs. To solve this problem, we introduce a maximal mean discrepancy (MMD) correction that penalizes differences between the probability distributions of internal activations of samples. Previous work has attempted batch correction by minimizing MMD. However, those models assume that batch effects

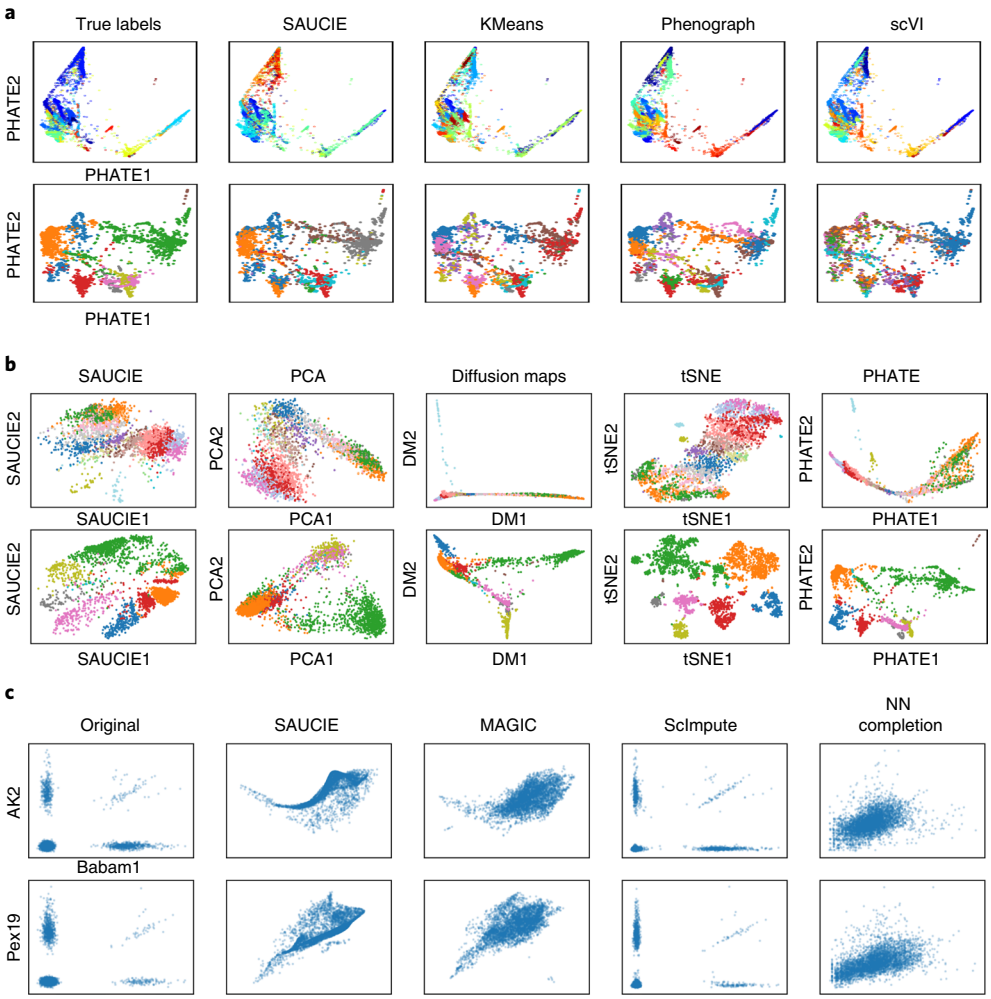

Fig. 3 | A comparison of the different analysis tasks performed by SAUCIE against other methods. a, A comparison of clustering performance on the data from Shekhar et al.[1] (top) and Zeisel et al.[2] (bottom) with sample sizes of 27,499 and 3,005, respectively. **b**, A comparison of SAUCIE's visualization on the same datasets as in **a**. **c**, A comparison of the indicated imputation methods on the 10x mouse dataset[3] subset of size 4,142.

are minor and simple shifts close to the identity function, which is often not the case[12]. Moreover, minimizing MMD alone removes any and all differences between batches. In contrast, the additional autoencoder reconstruction penalty in SAUCIE forces it to preserve the original structure in each batch, balancing the goals of, on one hand, making the two batches alike, while on the other hand not changing them. We note that this notion of a biological batch (data measured or run together) is distinct from the minibatches used in stochastic gradient descent to train neural networks. Here, the term batch is exclusively used to describe biological batches and when training with stochastic gradient descent the term minibatches is used.

Analyzing data before batch correction can lead to misleading results, as artificial variation from batch effects can drown out the relevant variation of interest within the biology (Fig. 4). Penalizing MMD directly on the input space would be a flawed way of addressing batch effects because it would require making the assumption of (and thus being sensitive to the choice of) meaningful distance and similarity measures on the input points. Since the data is noisy and possibly sparse, by instead penalizing MMD on an internal layer of the network, we can correct complex, highly nonlinear batch effects by aligning points on a data manifold represented in these layers.

Imputation and denoising. Next, we leverage the fact that an autoencoder does not reconstruct its input exactly, but instead must learn a lower dimensional representation of the data, and decode this representation for data reconstruction. This means the reconstructions are denoised versions of the input and are thus naturally solutions to the dropout and other noise afflicting much real-world data, especially single-cell RNA-sequencing data. The gene–gene relationships plotted in Fig. 3 illustrate the ability of SAUCIE to recover the meaningful relationship between genes despite the noise in the data. Thus, downstream activities such as differential gene expression are now enhanced by these improved expression profiles.

Visualization. Finally, we design the informational bottleneck layer of the autoencoder to be two-dimensional, which lets it serve as a visualization and nonlinear embedding of the data. Because the network must reconstruct the input accurately from this internal representation, it must compress all the information about a cell into just these two dimensions, unlike methods such as PCA or Diffusion Maps, which explicitly leave some variation unmodeled. Consequently, the information stored is also global, meaning points close together in the SAUCIE visualization are more similar than points that are farther apart, which is not true beyond small neighborhoods in a local method such as tSNE. The ability to flexibly learn and accurately reflect the structure in the data with SAUCIE is demonstrated in Fig. 3.

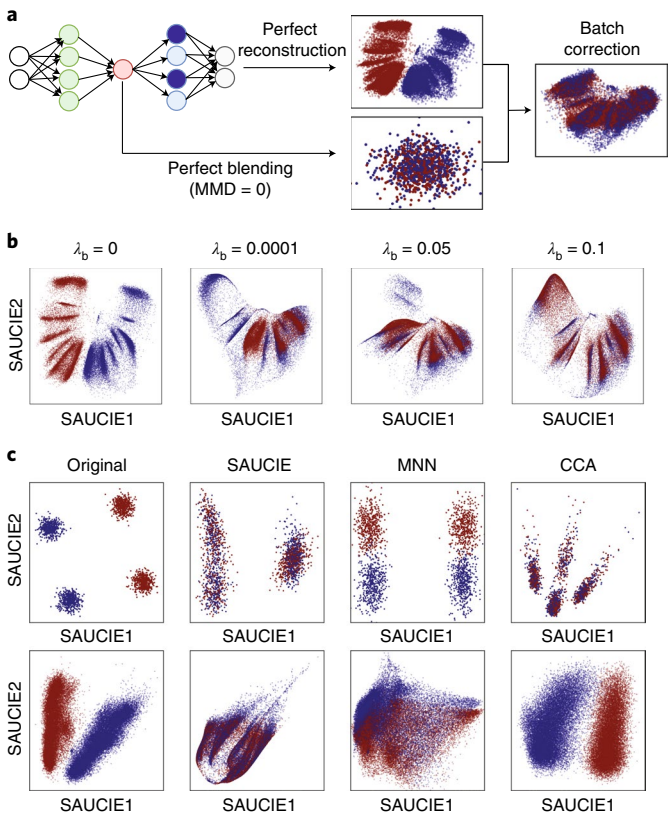

Fig. 4 | Demonstration of SAUCIE's batch correction abilities. a, SAUCIE batch correction balances perfect reconstruction (which would leave the batches uncorrected) with perfect blending (which would remove all of the original structure in the data) to remove the technical variation while preserving the biological variation. **b**, The effect of increasing λ_b (the magnitude of the MMD regularization) on the dengue data of size 41,721. **c**, Results of batch correction on the synthetic GMM data (of size 2,000) (top) and the dengue data (bottom) by the indicated methods.

Comparison to other methods. To assess how SAUCIE performs in relation to other methods on the tasks of clustering, visualization, batch correction, and imputation, we compare it to several of the leading methods dedicated to each task on ten public single-cell datasets. Five datasets are CyTOF: the dengue dataset we extensively evaluate below, T cell development data[13], renal cell carcinoma data[14], breast tumor data[15] and iPSC data[16]. Five datasets are scRNA-seq: mouse cortex data, retinal bipolar cells[17], hematopoiesis data[18], mouse brain data[19] and the 10x mouse megacell demonstration[20].

To evaluate SAUCIE quantitatively and qualitatively, we compare to methods dedicated to each of the four tasks: minibatch kmeans, Phenograph[21] and single-cell variational inference (scVI)[22] for clustering; mutual nearest neighbors (MNN)[23] and canonical correlation analysis (CCA)[24] for batch correction; PCA, Monocle2 (ref. [25]), diffusion maps, UMAP[26], tSNE[27] and PHATE[28] for visualization; MAGIC[29], scImpute[30] and nearest neighbors completion (NN completion) for imputation. SAUCIE performed better or as good as all of the methods by our quantitative metrics, despite most of the alternatives being designed for a single dedicated task, as opposed to SAUCIE being able to do all four tasks. Moreover, SAUCIE achieved its performance with greater scalability and faster runtimes than any of the other models (see Methods and Supplementary Figs. 1–6 for further details).

Analysis of immune response to dengue infection with SAUCIE. We illustrate SAUCIE's scalability with a large dataset that consists of single-cell CyTOF measurements of 11 million T cells from 45 subjects including a group acutely infected with the dengue virus and healthy controls from the same endemic area. This dengue data is an ideal test case for SAUCIE because the samples were collected over months, measured on different days and shipped from India. Thus, there is a pressing need for batch correction and data cleaning as well as uniform processing, clustering and meta-analysis of patient stratification.

Differential cluster proportions between subjects. We first batch correct and denoise the data using SAUCIE's MMD regularization, obtain the cluster characteristics of each group and then further analyze them for marker enrichments as single-cell versions of blood biomarkers[31]. To cluster the cells, we previously introduced two novel regularizations: ID regularization and within-cluster distance regularization. These are balanced against the autoencoder's reconstruction penalty with coefficients that control how much each part of the loss is weighted. We refer to these coefficients as λ_c and λ_d for ID regularization and within-cluster distance regularization, respectively. The two regularizations affect the number of clusters that result. For a given value of λ_d, as λ_c increases, the number of clusters decreases (coarser granularity). Higher values of λ_d yield more clusters (finer granularity). These two together act as knobs that can be tuned to get the desired granularity of clustering. For the clustering considered here, we use a coarse-grained clustering obtained with a λ_c of 0.1 and a λ_d of 0.2. If other granularities are desired, other coefficients could be used (Methods).

We look for T cell clusters that are differentially represented in the acute compared to the convalescent time points. These could be populations of cells with an important role in the infection or recovery process. We examine the marker abundance profile of each cluster to identify the subpopulation.

We find 20 total clusters within the T cell populations, five of which are CD8 T cells and 13 of which are CD4 T cells. In addition, there are six clusters of $CD4^-$ $CD8^-$ T cells, where four are relatively rare γδ T cells. These have been noted as a characteristic of reaction to viral infections[32–36]. There are 12 clusters representing effector memory cells and nine regulatory T cells that are $CD4^+$ $Foxp3^+$. Two of the clusters are naive T cells. Several of these populations are indicative of differences between acute, convalescent and healthy subjects, and can be used for characterizing the nature of the reaction of each of these groups. For example, an important but rare group of cells important in early immune response, γδ T cells, was identified and clustered (for further details, please see the Methods).

We can also visualize the cell-level cluster proportions on a patient manifold (Fig. 5). There, we see that cluster proportions arranged on this manifold reveal clusters that are changing across the space. This analysis indicates clearly that cluster 1 is representative of acute subjects and cluster 5 is representative of the healthy subjects. Furthermore, we can evaluate the same individual when measured after acute infection, and then later at a convalescent time point (Fig. 5). Viewed in this way, we see that cluster 11 is also more present in most subjects when they came in with an acute infection than at the convalescent time point.

Visualization. SAUCIE can process cells from all subjects to construct a cellular manifold and extract its features. First, we visualize this manifold using the two-dimensional visualization layer. Figure 6 is divided into two embeddings that show the cell manifolds for acute and healthy subjects separately. As can be seen, there is a characteristic change in the manifold that becomes apparent when comparing the embeddings side-by-side. The acute subjects have missing cell populations that are present in the healthy subjects, and vice versa.

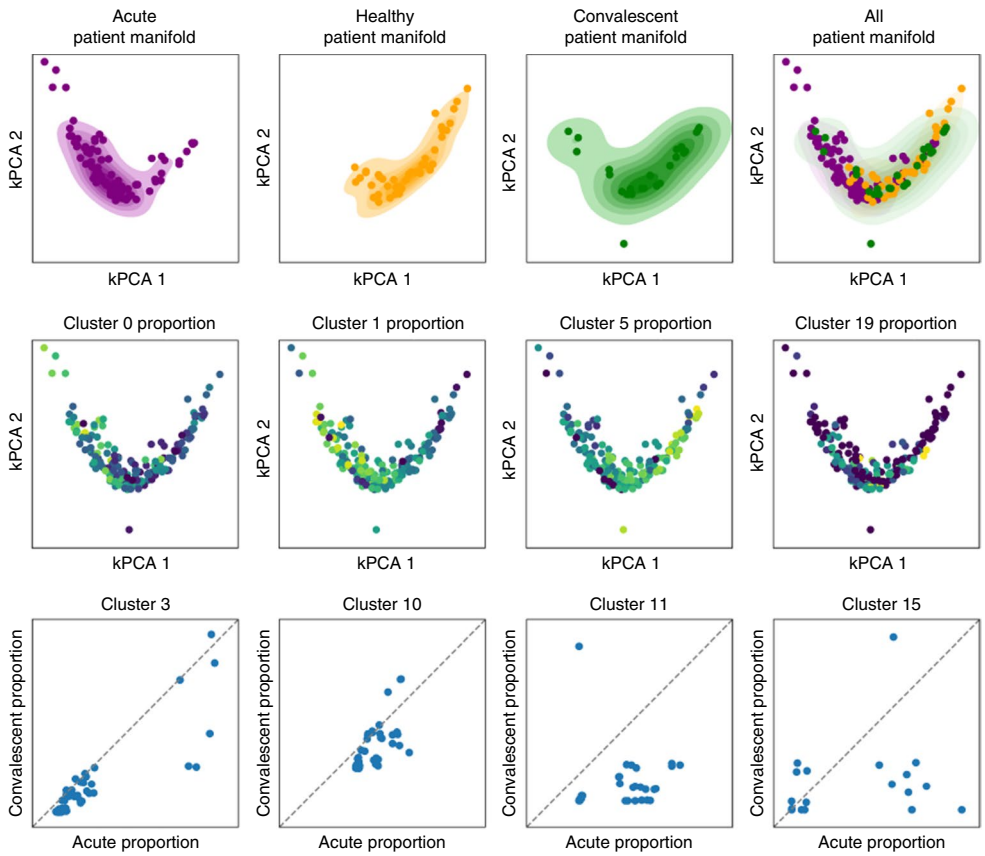

Fig. 5 | SAUCIE produces patient manifolds from single-cell cluster signatures. SAUCIE on the entire dengue dataset of 11,228,838 cells. Top row, the patient manifold identified by SAUCIE cluster proportions, visualized by kernel PCA with acute, healthy, convalescent and all subjects combined from left to right. Middle row, the same patient manifold shown colored by each patient's cluster proportion. Bottom row, a comparison of the cluster proportion for acute (*x* axis) versus convalescent (*y* axis) for patients that have matched samples.

After characterizing the nature of the cellular space in the aggregate, we can additionally analyze manifolds formed by the distributions of T lymphocytes within each patient separately (Fig. 5). As each patient has a heterogeneous population of cells, including with different total numbers of cells, it becomes a challenge to define a meaningful measure of similarity between the individuals. Here we are able to leverage the manifold constructed by the SAUCIE embedding and calculate MMD (a distribution distance) between the distribution of cells in the latent space for each pair of subjects. With a measure of similarity between each pair of patients, we can now construct a manifold not of the cells but also of the subjects.

Discussion

We presented SAUCIE, a neural network framework that streamlines exploratory analysis of datasets consisting of a multitude of samples and a large volume of single cells measured in each sample. The key advantage in SAUCIE is its ability to perform a variety of crucial tasks on single-cell datasets in a highly scalable fashion (using the parallelizability of deep learning with graphical processing units (GPUs)) without needing to call external algorithms or processing methods. As a result, SAUCIE is able to process multisample data in a unified way using a single underlying representation learned by a deep autoencoder. Thus, different samples can be visualized in the same coordinates without batch effects via the embedding layer of the neural network, and cluster proportions can be directly compared, since the whole dataset is decomposed into a single set of clusters without requiring cluster matching or metaclustering. These unified representations can be readily used for intersample comparisons and stratification, on the basis of their underlying cell-to-cell heterogeneity.

Mathematically, SAUCIE presents a new way of using deep learning in the analysis of biological and biomedical data by directly reading and interpreting hidden layers that are regularized in new ways to understand and correct different aspects of data. Thus far, deep learning has primarily been used in biology and medicine as a black-box model designed to train classifiers that often mimic human classifications of disease or pathology. However, the network internal layers themselves are typically not examined for mechanistic understanding. SAUCIE provides a way of obtaining information from internal layers of a deep network. Deep autoencoding neural networks essentially perform nonlinear dimensionality reduction on the data. As such they could be used 'off-the-shelf' for obtaining new coordinates for data in a reduced-dimension space, to which other algorithms can be applied. However, in SAUCIE we aim to go further to structure the reduced dimensions in specifically interpretable ways using new regularizations. Our information-theoretic regularization encourages near-binary activations of an internal layer, thus making the layer amenable to directly output encoded cluster identifications. We believe that such regularizations add interpretability to layers in neural networks, thus turning these 'black boxes' into 'glass boxes'.

Online content

Any methods, additional references, Nature Research reporting summaries, source data, statements of code and data availability and associated accession codes are available at https://doi.org/10.1038/s41592-019-0576-7.

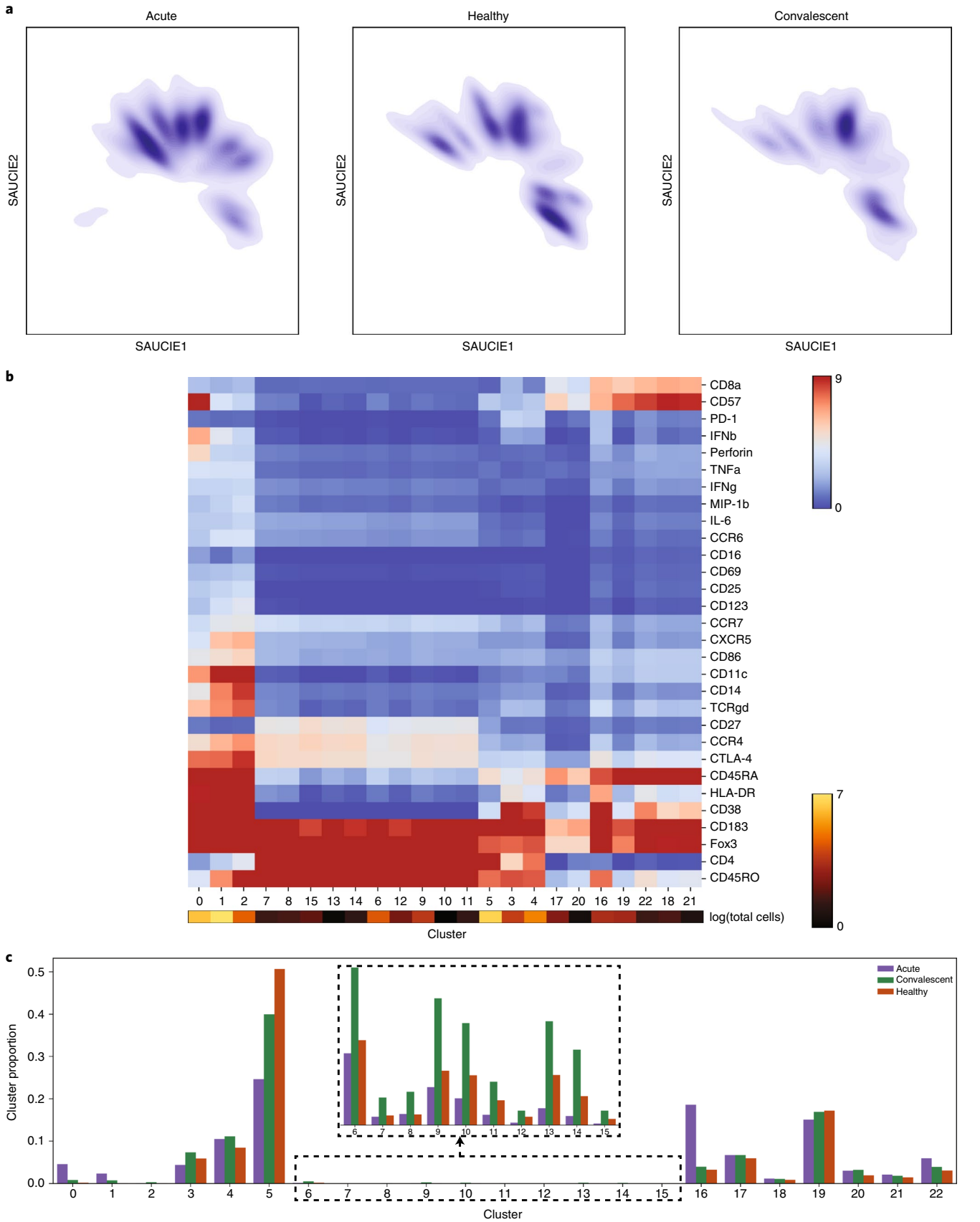

Fig. 6 | SAUCIE identifies and characterizes cellular clusters, whose proportions can be used to compare patients. SAUCIE on the entire dengue dataset of 11,228,838 cells. **a**, The cell manifolds identified by the two-dimensional SAUCIE embedding layer for the T lymphocyte subsets from acute, healthy and convalescent subjects. **b**, A heatmap showing clusters along the horizontal axis and markers along the vertical axis. Cluster sizes are represented as a color bar beneath the heatmap. **c**, Cluster proportions for acute, convalescent and healthy patients.

Received: 24 August 2018; Accepted: 19 August 2019;
Published online: 7 October 2019

References

1. Tan, J. et al. Unsupervised extraction of stable expression signatures from public compendia with an ensemble of neural networks. *Cell Syst.* **5**, 63–71 (2017).
2. Way, G. P. & Greene, C. S. Extracting a biologically relevant latent space from cancer transcriptomes with variational autoencoders. In *Pacific Symposium on Biocomputing 2018* Vol. 23 (PSB, 2018).
3. Wang, W., Huang, Y., Wang, Y. & Wang, L. Generalized autoencoder: a neural network framework for dimensionality reduction. In *CVPR Workshops* (eds Betke, M. & Davis, J.) 496–503 (IEEE, 2014).
4. Tan, J. et al. Unsupervised feature construction and knowledge extraction from genome-wide assays of breast cancer with denoising autoencoders. In *Pacific Symposium on Biocomputing 2015*. Vol. 20 (PSB, 2015).
5. Tan, J., Hammond, J. H., Hogan, D. A. & Greene, C. S. Adage-based integration of publicly available pseudomonas aeruginosa gene expression data with denoising autoencoders illuminates microbe-host interactions. *MSystems* **1**, e00025-15 (2016).
6. Chen, H., Shen, J., Wang, L. and Song, J. Leveraging stacked denoising autoencoder in prediction of pathogen-host protein-protein interactions. In *Proc. 2017 IEEE International Congress on Big Data (BigData Congress)* 368–375 (IEEE, 2017).
7. Chen, L., Cai, C., Chen, V. & Lu, X. Learning a hierarchical representation of the yeast transcriptomic machinery using an autoencoder model. *BMC Bioinforma.* **17**, S9 (2016).
8. Coifman, R. R. & Lafon, S. Diffusion maps. *Appl. Comput. Harmon. Anal.* **21**, 5–30 (2006).
9. Courbariaux, M., Bengio, Y. & David, J.-P. Binaryconnect: training deep neural networks with binary weights during propagations. In *Advances in Neural Information Processing Systems 28 (NIPS 2015)* (eds Cortez, C. et al.) 3123–3131 (JMLR, 2015).
10. Courbariaux, M., Hubara, I., Soudry, D., El-Yaniv, R. & Bengio, Y. Binarized neural networks: training deep neural networks with weights and activations constrained to +1 or −1. Preprint at https://arxiv.org/abs/1602.02830 (2016).
11. Tang, W., Hua, G. and Wang, L. How to train a compact binary neural network with high accuracy? In *Thirty-First AAAI Conference on Artificial Intelligence* (eds Singh, S. & Markovitch, S.) 2625–2631 (ACM, 2017).
12. Shaham, U. et al. Removal of batch effects using distribution-matching residual networks. *Bioinformatics* **33**, 2539–2546 (2017).
13. Setty, M. et al. Wishbone identifies bifurcating developmental trajectories from single-cell data. *Nat. Biotechnol.* **34**, 637 (2016).
14. Chevrier, S. et al. An immune atlas of clear cell renal cell carcinoma. *Cell* **169**, 736–749 (2017).
15. Azizi, E. et al. Single-cell map of diverse immune phenotypes in the breast tumor microenvironment. *Cell* **174**, 1293–1308 (2018).
16. Zunder, E. R., Lujan, E., Goltsev, Y., Wernig, M. & Nolan, G. P. A continuous molecular roadmap to ipsc reprogramming through progression analysis of single-cell mass cytometry. *Cell Stem Cell* **16**, 323–337 (2015).
17. Shekhar, K. et al. Comprehensive classification of retinal bipolar neurons by single-cell transcriptomics. *Cell* **166**, 1308–1323 (2016).
18. Paul, F. et al. Transcriptional heterogeneity and lineage commitment in myeloid progenitors. *Cell* **163**, 1663–1677 (2015).
19. Zeisel, A. et al. Cell types in the mouse cortex and hippocampus revealed by single-cell RNA-seq. *Science* **347**, 1138–1142 (2015).
20. *Single Cell Gene Expression Datasets* (10x Genomics, 2017); https://support.10xgenomics.com/single-cell-gene-expression/datasets
21. Levine, J. H. et al. Data-driven phenotypic dissection of AML reveals progenitor-like cells that correlate with prognosis. *Cell* **162**, 184–197 (2015).
22. Lopez, R., Regier, J., Cole, M., Jordan, M. & Yosef, N. A deep generative model for single-cell RNA sequencing with application to detecting differentially expressed genes. Preprint at https://arxiv.org/abs/1710.05086 (2017).
23. Haghverdi, L., Lun, A. T., Morgan, M. D. & Marioni, J. C. Batch effects in single-cell RNA-sequencing data are corrected by matching mutual nearest neighbors. *Nat. Biotechnol.* **36**, 421 (2018).
24. Butler, A. et al. Integrating single-cell transcriptomic data across different conditions, technologies, and species. *Nat. Biotechnol.* **36**, 411–420 (2018).
25. Qiu, X. et al. Reversed graph embedding resolves complex single-cell trajectories. *Nat. Methods* **14**, 979–982 (2017).
26. McInnes, L., Healy, J. & Melville, J. Umap: uniform manifold approximation and projection for dimension reduction. Preprint at https://arxiv.org/abs/1802.03426 (2018).
27. van der Maaten, L. & Hinton, G. Visualizing data using t-SNE. *J. Mach. Learn. Res.* **9**, 2579–2605 (2008).
28. Moon, K. R. et al. PHATE: a dimensionality reduction method for visualizing trajectory structures in high-dimensional biological data. Preprint at https://doi.org/10.1101/120378 (2017).
29. Van Dijk, D. et al. Recovering gene interactions from single-cell data using data diffusion. *Cell* **174**, 716–729 (2018).
30. Li, W. V. & Li, J. J. An accurate and robust imputation method scImpute for single-cell RNA-seq data. *Nat. Commun.* **9**, 997 (2018).
31. Regev, A. et al. Science forum: the human cell atlas. *eLife* **6**, e27041 (2017).
32. Panda, A. et al. Age-associated decrease in tlr function in primary human dendritic cells predicts influenza vaccine response.*J. Immunol.* **184**, 2518–2527 (2010).
33. Tsai, C.-Y. et al. Type I IFNs and IL-18 regulate the antiviral response of primary human γδ-T cells against dendritic cells infected with dengue virus. *J. Immunol.* **194**, 3890–3900 (2015).
34. Garcillán, B. et al. GD-T lymphocytes in the diagnosis of human T cell receptor immunodeficiencies. *Front. Immunol.* **6**, 20 (2015).
35. Chien, Y.-H., Meyer, C. & Bonneville, M. γδ-T cells: first line of defense and beyond. *Annu. Rev. Immunol.* **32**, 121–155 (2014).
36. Cimini, E. et al. Human Zika infection induces a reduction of IFN-γ producing CD4 T-cells and a parallel expansion of effector Vδ2 T-cells. *Sci. Rep.* **7**, 6313 (2017).

Acknowledgements

This research was supported in part by: the Indo-U.S. Vaccine Action Program, the National Institute of Allergy and Infectious Diseases of the NIH (Award no. AI089992 to R.R.M.); IVADO (L'institut de valorisation des données to G.W.) and the Chan–Zuckerberg Initiative (grant no. 182702 to S.K.).

Author contributions

M.A., S.K., G.W. and D.v.D. envisioned the project. M.A., K.S. and D.v.D., implemented the model and performed the analyses. M.A., S.K., G.W. and D.v.D. wrote the paper. K.S., W.S.C., H.M., A.C. and K.R.M. provided assistance in writing and analysis. Y.Z., X.W., M.V., A.D., V.R., P.K. and R.M. were responsible for data acquisition and processing.

Competing interests

The authors declare no competing interests.

Additional information

Supplementary information is available for this paper at https://doi.org/10.1038/s41592-019-0576-7.

Correspondence and requests for materials should be addressed to S.K.

Peer review information Nicole Rusk was the primary editor on this article and managed its editorial process and peer review in collaboration with the rest of the editorial team.

Reprints and permissions information is available at www.nature.com/reprints.

Publisher's note Springer Nature remains neutral with regard to jurisdictional claims in published maps and institutional affiliations.

Methods

Computational methods. In this section, we explain the SAUCIE framework in greater detail including the philosophy behind using autoencoders for learning the cellular manifold, details of the regularizations used in different layers of SAUCIE to achieve particular data analysis tasks as well as training and implementation details. Finally, we discuss the emergent higher-level organization of the patient manifold as a result of the cellular manifold of the subjects learned by SAUCIE.

Multitask manifold learning. A popular and effective approach for processing big high-dimensional data in genomics, as well as other fields, is to intuitively model the intrinsic geometry of the data as being sampled from a low-dimensional manifold—this is commonly referred to as the manifold assumption[37]. This assumption essentially means that local regions in the data can be linearly mapped to low-dimensional coordinates, while the nonlinearity and high dimensionality in the data comes from the curvature of the manifold. Typically, a notion of locality is derived from the data with nearest-neighbor search or adaptive kernels to define local neighborhoods that can approximate tangent spaces of the manifold. Then, these neighborhoods are either used directly for optimizing low-dimensional embeddings (for example, in tSNE and LLE), or they are used to infer a global data manifold by considering relations between them (for example, using diffusion geometry[8,28,29]).

The characterization of the intrinsic data geometry as a data manifold is also closely related to the underlying approach in SAUCIE. Indeed, neural networks can be considered as piecewise linear approximations of target functions[38]. In our case, we essentially approximate the data manifold coordinate charts and their inverse with the autoencoder architecture of SAUCIE. The encoder training identifies local patches and maps them to low-dimensional coordinates, while sewing these patches together in this embedding to provide a unified visualization. The decoder learns the linear relation between these intrinsic coordinates and the tangent spaces of the manifold, positioned in the high dimension. This also results in a projection of data points on the manifold (via its tangent spaces), which creates a denoising effect similar to the diffusion-based one used recently in MAGIC[29]. Finally, the clustering layer in SAUCIE is trained to recognize and aggregate similar data regions to ensure an appropriate granularity (or resolution) of the identified neighborhoods and prevent excessive fragmentation of the manifold.

While tools using the scaffold of manifold learning have emerged for various tasks in single-cell data analysis, there is currently no unified manifold model that provides all of the necessary tasks in a scalable fashion. For example, MAGIC uses manifold learning to impute the data, but does not address embedding, visualization or clustering. Diffusion pseudotime provides an organization of the data to infer latent temporal structure and identifies trajectories, but it does not deal with imputation, clustering or visualization. Furthermore, manifold learning methods do not work well across batches and typically just focus on single batches. Thus, their construction may suffer from batch effects and be dominated by the geometry between batches rather than their biology, as demonstrated by the example of Phenograph in Supplementary Fig. 1.

To address these shortcomings, SAUCIE performs all operations on a unified manifold geometry, which is learned implicitly by a deep multitasking neural network. It uses the scalability of deep learning to process high throughput data and construct a manifold that is jointly optimized for multiple tasks; namely, clustering, visualization, imputation and batch correction. Therefore, the tasks themselves respect the manifold assumption and have the associated advantages, such as robustness to noise, while also agreeing with each other on a coherent underlying structure of the data.

SAUCIE architecture. SAUCIE consists of three encoding layers, an embedding layer, and then three decoding layers. The default number of neurons per hidden layer in the encoder used were 512, 256 and 128 with a symmetric decoder. The Gaussian mixture model (GMM) dataset, being simpler, was clustered with layers of 50, 30 and 10. For batch correction, the best results were achieved with layer sizes of 1,024, 512 and 256. The ID regularization was applied to the final decoder layer, which uses a rectified linear unit. The two-dimensional embedding layer uses a linear activation, while all other layers use a leaky rectified linear activation with 0.2 leak. The coefficients λ_d and λ_c were chosen depending on the dataset, with the best values generally being λ_d twice λ_c. Their magnitude was guided by the effect of these two knobs on the granularity (shown in Supplementary Fig. 2). Training was performed with minibatches of 256, mean-squared error for the reconstruction error function, and the optimizer chosen is ADAM with learning rate 0.001.

Batch correction and MMD regularization. A principal challenge in the analysis of single-cell data is dealing with so-called batch effects that result from technical variability between replicates of an experiment. Combining replicates often results in technical and experimental artifacts being the dominant source of variability in the data, even though this variability is entirely artificial. This experimental noise can come in the form of dropout, changes of scale, changes of location or even more complicated differences in the distributions of each batch. It is infeasible to parametrically address all of the potential differences explicitly; for example, by assuming measurements are drawn from a Gaussian distribution. Instead of addressing specific explicit models of noise, SAUCIE minimizes a distance metric between distributions. The batch correction term L_b calculates the MMD between batches, as

$$L_b = \Sigma_{i \neq ref} \mathrm{MMD}(V_{ref}, V_i)$$

where V_{ref} is the visualization layer of one of the replicates, arbitrarily chosen to be considered as a reference batch. MMD compares the average distance from each point to any other point in its own batch, with the distance to points in the other batch, denoted V_i. MMD is zero only when two distributions are equal. Thus, minimizing this metric encourages SAUCIE to align the batches. MMD has been used effectively to remedy batch effects in residual networks, but here SAUCIE uses it in a feedforward autoencoder and combines it with other tasks of interest in biological exploratory data analysis[14].

The choice of reference does not affect the degree to which two distributions can be aligned, but a reference batch is necessary because the encoding layers of a standard network will be encouraged to embed different batches in different places in the visualization layer. It does this because the decoder is required to make its reconstruction $\hat{X}$ match the original data in X, which includes the batch effects. To remedy this, the decoder in SAUCIE is required to reconstruct the reference batch exactly as usual, but other batches must only be reconstructed to preserve the points normalized by mean and variance. Consequently, the MMD regularization term will be minimized when batches are aligned, and the decoder need only be able to reconstruct the exact values of the reference batch and the relative values of the nonreference batches. The nonreference batches will be aligned to the reference batch in a way that preserves their internal structure as best as possible.

Regularizations and post-processing for clustering. For ID regularization, we consider the task of clustering data points by interpreting the sparse layer B in the network as encoding cluster assignments. We note that a common activation function used to introduce nonlinearities in neural networks (including SAUCIE) is the rectified linear unit, and it provides a natural threshold for binarizing neuron activation to be either zero or one. These units are either 'off' at or below zero or 'on' for any positive value, so a small positive value ε can be used a threshold to binarize the activations in B. This results in an interpretable clustering layer that creates 'digital' cluster codes out of an 'analog' hidden layer, thus providing a binary code for each input point of the network. These binary codes are in turn used as cluster identifiers to group data points with the same code into a single cluster.

To automatically learn an appropriate granularity of clusters, we developed a novel regularization that encourages near-binary activations and minimizes the information (that is, number of clusters) in the clustering layer. Our regularization is inspired by the von Neumann (or spectral) entropy of a linear operator[39], which is computed as the Shannon entropy of their normalized eigenvalues. This entropy serves as a proxy for the numerical rank of the operator, and thus provides an estimation of the essential dimensionality of its range. In our case, we extend this notion to the nonlinear transformation of the neural network by treating neurons as our equivalent of eigenvalues, and computing the entropy of their total activation over a batch. We call this entropy ID and the corresponding ID regularization aims to minimize this entropy while still encoding sufficient information to allow reconstruction of the input data points.

The ID regularization is computed from the clustering layer activations in B by first computing the activation of each neuron j with each connection from the previous layer i as $a_j = \sum_{i=1}^{n} B_{ij}$, with n being the total number of neurons in the previous layer and k being the number of neurons in the clustering layer, then normalizing these activations to form an activation distribution $\mathbf{p} = \frac{\mathbf{a}}{\|\mathbf{a}\|_1}$ and finally computing the entropy of this action distribution as

$$L_c(B) = -\sum_{j=1}^{k} p_j \log p_j$$

By penalizing the entropy of neuron activations, this regularization encourages a sparse and binary encoding. This counters the natural tendency of neural networks to maximize the amount of captured (that is, encoded) information by spreading activations out across a layer evenly. By forcing the activations to be concentrated in just a few distinct neurons, different inputs end up being represented with rather similar activation patterns and thus naturally clustered. When combined with the reconstruction loss, the network will retain enough information in the sparse layer for the decoder to reconstruct the input, keeping similar points in the same cluster.

Intracluster distance regularization. The digital codes learned by SAUCIE create an opportunity to interpret them as clusters, but these clusters would not necessarily comprise only similar points. To emphasize that inputs only be represented by the same digital code if they are similar to each other, SAUCIE also penalizes intracluster pairwise distances. Beyond suffering reconstruction loss, using the same code for points that are far away from each other will now incur an even greater loss.

This loss is calculated as the Euclidean distance between points with the same binary code:

$$L_d(B,\hat{X}) = \sum_{i,j:b_i=b_j} ||\hat{x}_i - \hat{x}_j||^2$$

where $\hat{x}_i$, $\hat{x}_j$, b_i, b_j are the ith and jth rows of $\hat{X}$and B, respectively.

Since ID regularization is minimized by using the same code to represent all inputs, this term acts as an opposing balance. Intracluster distances are minimized when all points are in a cluster by themselves. Together with the reconstruction penalty, these terms encourage SAUCIE to learn clusters that are composed of as many points as possible that are near to each other.

An additional benefit of clustering via regularization is that not only is the number of clusters not needed to be set a priori, but by changing the value of λ_c the level of granularity of the clustering can be controlled, so both coarse clustering and fine clustering can be obtained to further add insight into the underlying structure of the data.

Cluster merging. As the binarized neural network may not converge to the ideal level of granularity due to the many possible local optima in the loss landscape, we process the SAUCIE clustering with a cluster merge step to fix the ideal level of granularity everywhere. The cluster merging is performed by calculating MMD between clusters in the SAUCIE latent space and merging all clusters $i, j \in C$, where C is the set of all clusters, such that both of the following equations hold

$$\underset{\xi \in C}{\text{argmin}} \text{MMD}(i,\xi) = j$$

$$\underset{\xi \in C}{\text{argmin}} \text{MMD}(j,\xi) = i$$

This merging finds clusters that would be a single cluster in another granularity and fixes them to a single cluster.

Patient manifold visualization. In addition to the cell-level manifold constructed by SAUCIE, we also consider the geometry between samples to provide a coarser patient-level manifold. We construct and embed this manifold in low dimensions by applying kernel PCA (kPCA) with an radial basis function kernel to the metric space defined by MMD distances between subjects. This augments the analysis SAUCIE provides of the biological variations identified in the cell space with an analysis of the variation in the patient space. Normally, without batch correction, the two sources of variation would be confounded and batch effects would prevent clear analysis at either level (patient or cell) across batches. With our approach here we are able to separate them to provide on one hand, a stable (batch-invariant) cell-level geometry by the SAUCIE embedding, and on the other hand, a robust patient geometry provided by kPCA embedding. The patient geometry then allows us to recover patient-level differences and use them further for data exploration, in conjunction with the cell-level information. For example, as Fig. 5a shows, we have a notable stratification between the acute and nonacute subjects. There is also a noticeable difference between the convalescent subjects and the acute, albeit a less drastic one than the difference between acute subjects and the others.

Training. To perform multiple tasks, SAUCIE uses a single architecture as described above, but is run and optimized sequentially. The first run imputes noisy values and corrects batch effects in the original data. This preprocessed data is then run through SAUCIE again to obtain a visualization and to pick out clusters. The different runs are done by optimizing different objective functions. In the following, we describe the optimization of each run over a single batch of n data points. However, the full optimization of each run independently uses multiple (mini-)batches to converge and minimize the described loss functions. For the first run, formally let X be an $n \times d$ input batch, where each row is a single data point and d is the number of features in the data. It is passed through a cascade of encoding linear and nonlinear transformations. Then, a cascade of decoding transformations reconstruct the denoised batch $\hat{X}$, which has the same dimensions as the input X and is optimized to reconstruct it.

For the next run, the cleaned batch $\hat{X}$ is passed through encoding transformations and a visualization layer denoted by $V \in R^{n \times 2}$. We also consider a clustering layer in another run where the decoder outputs near-binary activations $B \in R^{n \times d_B}$, where d_B is the number of hidden nodes in the layer, which will be used to encode cluster assignments as described below. The activations in B are then passed to the reconstruction $\tilde{X}$ that has the same dimensions as $\hat{X}$ (and X) and is optimized to reconstruct the cleaned batch.

The loss function of all runs starts with a reconstruction loss L_r forcing the autoencoder to learn to reconstruct its input at the end. SAUCIE uses the standard mean-squared error loss. We note that while mean-squared error is a standard and effective choice in general, other loss functions can also be used here as an application-specific substitutes that may be more appropriate for particular types of data. For the first run, we add to this loss a regularization term L_b that enables SAUCIE to perform batch correction. This regularization is computed from the visualization layer to ensure consistency across subsampled batches. The resulting total loss is then

$$L = L_r(X,\hat{X}) + \lambda_b L_b(V)$$

The loss function of the clustering run then optimizes L_r along with two regularization terms L_c and L_d that together enable SAUCIE to learn clusters:

$$L = L_r(\hat{X},\tilde{X}) + \lambda_c L_c(B) + \lambda_d L_d(B,\tilde{X})$$

The first term L_c guides SAUCIE to learn binary representations via the activations in B using a novel information dimensionality penalty that we introduce in this paper. The second term L_d encourages interpretable clusters that contain similar points by penalizing intracluster distances in the cleaned batch $\hat{X}$, which is fixed for this run.

Dengue dataset batch correction. Beyond the sheer size of the total dataset, due to the large number of distinct samples in the experiment there are significant batch-related artifact effects, stemming from day-to-day differences, instruments, handling and shipping of the samples. While there are true biological differences between the individual samples, to identify those true differences in the samples we have to remove differences that are caused by these technical variables.

Differences that are highly associated with the day the samples were run on the cytometry instrument can be seen by grouping all of the samples together by run day and examining their marker-by-marker abundances. Each run day has 12 samples chosen such that each day has samples from each experimental condition, so any differences between the samples from each day are batch effects. As shown in Supplementary Fig. 3, these differences exist in the spike-in controls as well as the samples, confirming their identity as batch effect and not true variation.

Supplementary Fig. 4 shows four markers with extreme batch effects: TCRgd, IL-6, IFNg and CD86. These batch effects would normally mean only samples within each run day could be compared to each other, as comparisons between samples from different run days would be dominated by the differences in the run days. Instead, the SAUCIE batch correction removes these undesirable effects by combining the samples from each day and aligning them to a reference batch, here chosen to be Day 1. Supplementary Fig. 4 shows that after SAUCIE the differences between run days disappear so that now what it means to be low or high in a marker is the same for each day. Before, the cells with the lowest IFNg in samples from Day 3 would still be considered $IFNg^+$ while the cells with the highest IFNg in samples from Day 1 would still be $IFNg^-$. After batch correction with SAUCIE, these can be directly compared.

The challenge of batch correction is to remove differences due to artifacts while preserving biological differences. We reason that to prevent removing true biological variation, the 'shape' of the data (but not its position and scale) within each day must be preserved. We define the shape of the data as any moment beyond the first two: mean and variance. We examine this in detail by considering a run day with the most significant batch effects, Day 2. In Supplementary Fig. 3c, the SAUCIE visualization shows that the reference and nonreference batches are completely separated. When MMD regularization is added in SAUCIE, however, these two batches are fully overlapped. In Supplementary Fig. 5, we examine the 12 individual samples that were run on Day 2. Initially, we see that this confirms our idea that the differences between days are batch effects, because each sample measures high in IL-6 and CD86. So the differences between samples run on Day 1 and Day 2 in CD86 abundance is not dominated by having more of a certain sample type in Day 2. Instead, all samples in Day 2 have been shifted higher. As desired, after batch correction, the mean of each marker is reduced to the level of the reference-batch mean. Crucially, the relationship of samples in Day 2 relative to each other is preserved. The samples with the highest IL-6 in Day 2 are still Samples 3, 9 and 11 while the samples with the lowest are still Samples 4, 5 and 6. SAUCIE has just changed what it means to be high or low for samples in this day such that it reconciles what it means to be high or low for samples in the reference day.

Dengue cluster details. Here, we detail specific clusters found in our SAUCIE analysis of the dengue data.

1. γδ T cells are a relatively rare type of T cells that SAUCIE successfully identifies. Despite their rarity, they appear to have significance in identifying different populations, which emphasizes the importance of this attribute of SAUCIE. These cells signal especially strong early in immune response, particularly skin and mucosal immunity. They have less variable TCR sequences than αβ T cells[40]. These cells are a bridge between T cells and myeloid cells, as they have some innate immune activity, where they express CD11c and CD86. They can bind to lipid antigens. Clusters 0 and 3 (consisting of 7% of the total cells) show upregulation of CD57. This is an indication of terminal differentiation. CTLA-4 and CD38 are also high, so these are highly activated cells and potentially dysfunctional. We see that these clusters have the highest proportion in the acute subjects and lowest in the healthy subjects. Out of the 15 subjects that were measured both as acute subjects and later in convalescence, 13 had more of these cells during their acute infection.

2. We find another group of γδ T cells that are CD45RO and CD45RA positive (cluster 2, consisting of 1% of the total cells), but not yet fully terminally differentiated, so these could be transitional between naive and effector memory. The effector memory cells express less IFNb. As this cluster is more expressed in the healthy subjects, it indicates that even these subjects may have had some exposure to dengue. There is a lack of an inflammatory state, that is, low in IFNb and Perforin, so we expect that these are actually memory cells instead of effector cells. It makes sense, then, that these populations are more expressed in convalescent and healthy subjects.
3. We also find another population of $CD4^+$ T cells (clusters 3–15, consisting of 45% of the total population) that are not expressing any inflammatory markers or activation markers, and these are higher in the convalescent and healthy subjects, while being very low in the acute subjects. These cells look to be other memory cells that may characterize these convalescent subjects. In fact, out of the 15 subjects with acute-convalescent paired measurements, 11 had more of these cells during convalescent measurement. These have signs of recent activation as they do not have CD69, which is an early activation marker, or any of the cytokines such as IFNg, IFNb or IL-6.
4. Additionally, we find a population of $CD8^+$ effector cells (cluster 15, which consists of 3% of the total cells) that are highly expressed in the acute subjects. These cells also express CD57 and CD38, but are not γδ as the previous populations were. These appear to be more differentiated and are likely not transitional either, as the previous ones were.

Comparison to phenograph. We next compare the SAUCIE pipeline of batch correcting, clustering and visualizing single-cell data from a cohort of subjects to an alternative approach called metaclustering[21]. We first cluster each sample individually with Phenograph. Then, we represent each cluster as its centroid and use Phenograph again on the clusters to obtain metaclusters. We examine the pipelines on ten of the 180 samples here, where the metaclustering approach took 40 min. We note that the SAUCIE pipeline took 45 min to process all 180 samples, while the metaclustering approach would take 12 h to process all of them. Supplementary Fig. 1 shows tSNE embeddings of the cluster centroids where the size of the cluster is proportional to the size of the point. Coloring by sample, we see that the metaclusters have identified batch effects. Metacluster 0 is only composed of samples 1, 3, 4 and 5. These samples have no clusters in any other metacluster, and none of the other samples have any cluster in this metacluster. Examining the gene expression heatmap, we see that metacluster 0 has separated cells with high CD86 values, which were shown earlier to be batch effects. Moreover, the metaclusters are very heterogeneous internally with respect to gene expression. This is a result of metaclustering the cluster centroids, as the metaclusters then have no information about the individual cells comprising that centroid.

In contrast, Supplementary Fig. 6 shows the SAUCIE pipeline on these ten samples. The cluster proportions show that each cluster is fully mixed with respect to the samples, as opposed to the sample-segregated metaclusters of the previous approach. Similarly, the clusters are more homogeneous internally, meaning they actually keep similar cells together, as opposed to the metaclusters, which lost this information when each cluster was represented by only its centroid. Finally, we find that SAUCIE effectively compares cells across subjects, while the metaclustering approach still fails at patient-to-patient comparisons, instead only identifying batch effect variation. This emphasizes the importance of multitask learning using a unified representation in SAUCIE.

Comparison to other methods. Here, we present the full detailed analysis of the comparison to other methods referenced in the main text.

Clustering. To evaluate the ability of SAUCIE to find meaningful clusters in single-cell data, we compare it to several alternative methods: minibatch kmeans, Phenograph and another neural network approach called single-cell variational inference (scVI). While we compare to scVI as it and SAUCIE are both neural networks, we emphasize a fundamental difference between the two: scVI only returns a latent space, which must then be visualized or clustered by another outside method, while SAUCIE explicitly performs these tasks. Since kmeans needs to be told how many clusters there are ahead of time (k), we use the number of clusters identified by Phenograph as k. We look at the following datasets: artificially generated Gaussians rotated into high dimensions, and public single-cell datasets for which we have curated cell clusters as presented by the authors.

In addition to analyzing the clusters visually (Supplementary Fig. 7), we also quantitatively assess cluster performance of the methods by computing modularity and silhouette scores on the generated clusters and ground truth labels (Supplementary Table 1). First, we look at an artificially generated dataset of four two-dimensional Gaussian point clouds with different means rotated into 100 dimensions. We find that SAUCIE is the only method that automatically identifies exactly four clusters, which was the underlying number of clusters in the generation model. This illustrates why optimizing modularity, such as Phenograph does, is not necessarily the best heuristic to follow, as it adds additional complexity to the clustering to increase the modularity score, resulting in too many clusters. Likewise, scVI did not identify the four clusters, which is unsurprising as the data did not fit its parametric model appropriate for gene counts.

We also examine clustering performance on five public single-cell datasets to evaluate the ability of SAUCIE to cluster real biological data. Visual inspection reveals that SAUCIE produces clusters that are qualitatively coherent on the embedding. Quantitatively, the modularity scores of its clusters corroborate this evaluation. As shown in Supplementary Table 1 the average modularity score across datasets is 0.8531. In a wide variety of data from both CyTOF and scRNA-seq measurements, SAUCIE is able to produce clusters that reasonably represent the data qualitatively, quantitatively and by comparison to other methods.

Batch correction. We assess our ability to remove batch-related artifacts with SAUCIE by comparison to two published batch correction methods that have been specifically designed to remove batch effects in single-cell data. The first, MNN, uses mutual nearest neighbors on a k-nearest neighbors graph to align two datasets, and the second, CCA, finds a latent space in which the two batches are aligned. To evaluate the performance of these methods and SAUCIE, we use several different datasets with varying degrees of batch artifacts. We note that SAUCIE is the only method capable of scaling batch correction to hundreds of samples as we do in the next section. Nonetheless, here we compare performance on datasets small enough for the alternative methods to handle.

To quantitatively assess the quality of batch correction, we apply a test we term the mixing score[41]:

$$\text{mixing score} = \frac{N_{b1}}{N_{b2}} \sum_{xj \in \text{KNN}(x_i)} (1_{\text{batch}(x_i)=\text{batch}(x_j)}) \quad (1)$$

where N_{b1} and N_{b2} are the number of points in the first and second batch, respectively. This score calculates for each point the number of nearest neighbors that are in the same batch as that point, accounting for the difference in batch sizes. In perfectly mixed batches, this score is 0.5, while in perfectly separated batches it is 1.0. As batch correction should not only mix the batches but also preserve their shape as best as possible, we quantify the distortion between the original and batch-corrected data using Procrustes, which finds the error between the optimal alignments of the two batches by linear transformation[42]. These numbers are reported in Supplementary Table 2. While the other methods each have some datasets that violate their assumptions and thus they perform poorly, SAUCIE performs as well or better at each of the wide variety of datasets.

We compare SAUCIE to the alternative methods on artificial GMM data, spike-in CyTOF samples from the dengue dataset, nontechnical scRNA-seq replicates from developing mouse cortex and four public datasets. In Supplementary Table 2, we see SAUCIE has the best average mixing score across the wide range of data types, without distorting the data more than is necessary.

Visualization. To evaluate the SAUCIE visualization and its ability to provide a faithful low-dimensional data representation, we provide comparisons of this visualization to other frequently used methods. We make use of artificial datasets where the underlying structure is known, as well as real biological datasets that have been extensively characterized previously, so we have previous understanding of the structure we expect to see in the visualization (Supplementary Fig. 8).

We measure the quality of the visualizations with a quantitative metric[43]. In line with their method's precision and recall metrics, we compute a neighborhood around each point in both the original data space and the embedding space and compare the neighbors of each. An embedding with high recall has most of a point's original-space neighbors in its embedding-space neighborhood. Similarly, an embedding with high precision has most of the point's embedding-space neighbors in its original-space neighborhood. As directed by the authors' algorithm, we gradually increase the size of the neighborhood and report the area-under-the-curve for the precision-recall curve. These results are in Supplementary Table 3, where SAUCIE has the highest average score of 0.9342, averaged across all datasets. Despite this, we note that this is only a heuristic and can give undesirable results at times, as it only looks at fixed neighborhoods in the original input space. For example, PCA fails to produce any visual separation in the data from Zunder et al., yet scores well by this metric. Likewise, tSNE artificially shatters the trajectories in the diffusion limited aggregation data, yet produces a high score. Nonetheless, this metric offers some corroboration at the quality of SAUCIE's visualization, a hard task to measure quantitatively.

In Supplementary Fig. 8, we see some methods preserve global information, but frequently at the expense of not preserving local and more fine-grained variation, such as PCA. Other methods, such as Diffusion Maps, provide visualizations that look like connected trajectories on every dataset, no matter the underlying distribution. tSNE preserves local information, but at the expense of not preserving global information, on the other hand. SAUCIE, meanwhile, balances global and local information preservation and provides varied visualizations, depending on the structure of the underlying data.

Imputation. We analyze the SAUCIE imputation and its ability to recover missing values by implicitly interpolating on a data manifold in several ways. First, Supplementary Fig. 9 shows several relationships from the scRNA-seq data of the

10x mouse megacell dataset affected by severe dropout. This dataset consists of 1.3 million cells, and SAUCIE was the only method in the comparison to be able process the full dataset. Moreover, it was able to do this in just 44 min. Additionally, because training a neural network only requires small minibatches in memory at one time, we were able to do this without ever loading the entire large dataset into memory all at once. Thus, to enable this comparison, we subsampled the data by taking one of the SAUCIE clusters consisting of 4,172 cells.

For this comparison, we measure against several popular imputation methods for scRNA-seq data: MAGIC, which is a data diffusion based approach, scImpute, which is a parametric statistical method for imputing dropouts in scRNA-seq data, and NN completion, which is an established method for filling in missing values in a general application of high-dimensional data processing.

In Supplementary Fig. 9, we show six relationships of the mouse megacell dataset for the original data and the different imputation methods. We observe that the original raw data is highly sparse, which can be seen by the large number of values on the axes where one of the variables is exactly zero. Note that most cells have one or both genes missing. This is a problem because this prevents us from identifying trends that exist between the genes. After imputation with SAUCIE, we can observe that the sparse character of the data has been removed, with values filled in that reveal underlying associations between the gene pairs. These associations are corroborated by MAGIC, which imputes similar values to SAUCIE in each case. MAGIC is a dedicated imputation tool that is widely used, so SAUCIE matching the relationships it found gives confidence in the ability of SAUCIE to impute dropout effectively. The resulting imputation in scImpute does not look significantly less sparse from the original and we do not see continuous trends emerge. NN completion appears to desparsify the data, but the resulting trends all look similar to each other (that is, positively correlated). This suggests that it does not correctly identify the underlying trends, as we would expect different genes to have different relationships. While scRNA-seq is highly sparse, the undersampling affects all entries in the matrix, including the nonzero values. As such, manifold-based methods such as SAUCIE and MAGIC are more suited for finding these true relationships because they denoise the full dataset as opposed to just filling in zeros.

Due to the fact that ground truth values for the missing counts in this single-cell data are not known, we further test the accuracy of the imputation abilities of SAUCIE with an artificially constructed experiment. We first leverage the bulk RNA-sequencing data of 1,076 cells[44], because it accurately captures the relationships between genes due to it not being sparse (as opposed to generating our own synthetic data from a parametric generating function that we have the ability to choose, where we can create the relationships). We then simulate increasing amounts of dropout and compare the imputed values returned by each method to the true values we started with. To simulate dropout in a manner that reflects the underlying mechanisms of inefficient messenger RNA capture, we remove molecules instead of just setting values for genes to zero. As a result, the level of dropout is conditional on the expression level, reflecting the dropout structure of scRNA-sequencing data. The results are reported in Supplementary Fig. 10, where SAUCIE compares favorably to other methods, recovering the true values accurately even after as much as 99% dropout. The dataset for this experiment consisted of just 1,076 cells, which allowed us to compare to the methods that cannot process larger datasets, but even on a dataset of this size SAUCIE gave a more than 100-times speedup over NN completion and 600-times speedup over scImpute.

Runtime comparison. To showcase the scalability of SAUCIE, we compare to a host of other methods on a subset of our newly generated CyTOF dataset consisting of over 11 million cells existing in 35 dimensions. We display the runtimes of each method on a random sample of N points, with $N = 100, 200, 400, 800, \ldots, 11{,}000{,}000$ in Supplementary Fig. 11. For each step, the method was given a timeout after 24 h. Points where a method stopped scaling in Supplementary Fig. 11 are marked with an 'x'.

SAUCIE performs visualization, batch correction, imputation and clustering in its run, while each of the other methods only performs one of these tasks. Moreover, SAUCIE does not just compute simple linear functions on the data, but instead performs complex nonlinear transformations in the process. Despite its complexity, it also scales very well with the extremely large dataset sizes, which can be further improved by simply adding more independent GPUs for calculations. Each additional (relatively inexpensive) GPU can offer a near linear increase in computation time, as opposed to more central processing units that offer diminishing returns in parallelizability. All experiments were run on a single machine with just one GPU, meaning these results could still benefit even more from this potential for scalability. For further details on how the runtime experiment was performed, see the Methods section.

Among the batch correction methods, there are no other methods that correct multiple batches simultaneously. However, even when we restrict to pairwise comparisons, SAUCIE is the only method that comes close to handling this amount of data. CCA and MNN both stop scaling in the tens of thousands of cells. In the group of imputation methods, scImpute and NN completion also stop scaling in the tens of thousands, while MAGIC stops scaling in the hundreds of thousands. For visualization, PCA was the only method faster than SAUCIE, which is unsurprising because calculating it using fast randomized singular value decomposition is quick, but it gives a simple, strictly linear blurry views of the data, in contrast to SAUCIE's nonlinear dimensionality reduction. The other more complex visualization methods do not scale to these dataset sizes: Diffusion Maps, PHATE, tSNE and Monocle2 all stop scaling before even reaching the full 11 million cells. For clustering, kmeans is the only one faster than SAUCIE, due to using its minibatched version. However, it still assumes circular clusters in the Euclidean space and comes with the intrinsic flaw that the number of clusters must be known ahead of time, which is not possible in any realistic setting such as ours where we are performing exploratory data analysis on a large new dataset. Phenograph and scVI also do not scale to the full dataset. Despite being another neural network method, scVI cannot scale to these larger sizes because it only produces a latent space that then must be clustered with another method. This requirement then becomes its bottleneck, emphasizing the importance of SAUCIE performing all tasks directly instead of acting as a pre-processing step for other methods.

SAUCIE is the only method that can efficiently batch correct, impute and denoise, visualize and cluster datasets of this size, while using a nonlinear manifold representation of the data.

Runtime comparison methodology. For each visualization, clustering, and imputation method, the dataset of size N was given to the method as input and returned the appropriate output. For batch correction, the dataset of size N was divided into two equal-sized batches that were corrected. For the methods that operated on minibatches, minibatches of size 128 were used. For the methods that train by stochastic gradient descent, the number of steps was determined by taking the total number of points and dividing by the size of the minibatch, so that a complete pass through the entire dataset was performed. To return clusters, the latent space of scVI must be clustered by another method, and since the number of clusters is not known ahead of time, the fastest method that does not require this to be known (Phenograph) was used. For SAUCIE, batch correction, imputation, clustering, and visualization were all produced in the timed run. All computations were performed on a single machine with 16 central processing unit cores and a GeForce GTX 1080 GPU.

Number of clusters. As discussed earlier, the number of clusters resulting from SAUCIE is not specified in advance, but dictated by the structure of the data that the model discovers, and by the choice of regularization coefficients λ_d and λ_c. For a given value of λ_d, as λ_c increases, the number of clusters decreases. Increasing λ_d, on the other hand, increases the number of clusters (Supplementary Fig. 2). This is because λ_c penalizes entropy in the activations of the n neurons in the clustering layer of the network. While entropy can be initially decreased by making all n neurons either 0 or 1, it can be further decreased by making all n neurons 0. Thus, as this term is considered more influential in the total loss, in the extreme, all points can be mapped to the same binary code. In contrast, λ_d penalizes intracluster distances, so this value can be decreased by making clusters smaller and smaller (and thus getting more of them). In the extreme for this term, every point can be made its own cluster and intracluster distances would decrease to 0. By balancing these two, the desired granularity of clustering can be obtained from SAUCIE. In our experiments, we find making λ_d to be between two and three times larger than λ_c, with values around 0.2 generally results in medium coarse-grained clustering. Another consideration that affects the number of clusters is the number of neurons in the clustering layer. We found varying this number does not improve performance and for all experiments here we use a fixed size of 256 neurons.

Experimental methods. *Study subjects.* Patients with dengue virus infection and healthy volunteers were enrolled with written informed consent under the guidelines of the Human Investigations Committees of the NIMHANS and Apollo Hospital, and Yale University. The Human Investigations Committee of each institution approved this study. Patients with dengue virus infection were defined as having dengue fever using WHO-defined clinical criteria, and/or laboratory testing of viral load or serotyping at the time of infection. Healthy volunteers included household contacts of patients with dengue virus present in the same endemic area. Participants were of both genders (26.7% female) and were all of Indian heritage. Subjects from the symptomatic and healthy groups were not statistically different for age, gender or race in this study.

Sample collection and cell isolation. Heparinized blood was collected from patients and healthy volunteers and employed a 42 marker panel of metal-conjugated antibodies following methods previously described[45,46]. Purification of peripheral blood mononuclear cells (PBMCs) was performed by density-gradient centrifugation using Ficoll-Paque (GE Healthcare) according to the manufacturer's instructions following isolation and cryopreservation guidelines established by the Human Immunology Phenotyping Consortium. PBMCs for CyTOF were frozen in 90% FBS containing 10% DMSO and stored in liquid N_2 for shipping following the guidelines of the Department of Biotechnology. Samples for this study were received in three shipments and viability was average 85% (range 50–98) across the dates.

Mass cytometry acquisition. For mass cytometry at Yale University, PBMCs (5×106 cells per vial) were thawed incubated in Benzonase (50 U ml^{-1}) in RPMI/10% human serum, and seeded in 96-well culture plate ($6\times103-1.2\times106$ cells per well. Monensin (2 µM, eBioscience) and Brefeldin A (3 µg ml^{-1}, eBioScience) added for the final 4 h of incubation for all groups. Groups of samples (8–13 d) were infected in vitro per day on five separate days and included a CD45-labeled spike-in reference sample in every sample. Surface markers were labeled before fixation and detailed staining protocols have been described. Briefly, cells were transferred to 96-well deep well plates (Sigma), resuspended in 25 µM cisplatin (Enzo Life Sciences) for 1 min and quenched with 100% FBS. Cells were surface labeled for 30 min on ice, fixed (BD FACS Lyse), and frozen at −80 °C. Intracellular labeling was conducted on batches of cells (12 per day). Fixed PBMCs were permeabilized (BD FACS Perm II) for labeling with intracellular antibodies for 45 min on ice. Cells were suspended overnight in iridium interchelator (125 nM; Fluidigm) in 2% paraformaldehyde in PBS and washed 1× in PBS and 2× in H_2O immediately before acquisition. A single batch of metal-conjugated antibodies was used throughout for labeling panels. Metal-conjugated antibodies were purchased from Fluidigm, Longwood CyTOF Resource Core or carrier-free antibodies were conjugated in house using MaxPar X8 labeling kits according to the manufacturer's instructions (Fluidigm). A total of 180 samples were assessed by the Helios (Fluidigm) on 15 independent experiment dates using a flow rate of 0.03 ml min^{-1} in the presence of EQ Calibration beads (Fluidigm) for normalization. An average of 112,537 ± 71,444 cells (mean ± s.d.) from each sample were acquired and analyzed by CyTOF. Data was preprocessed with the hyperbolic sine transformation. Additional experimental details will be given[10].

Reporting Summary. Further information on research design is available in the Nature Research Reporting Summary linked to this article.

Data availability

Data for the dengue dataset is available at Cytobank, with accession number 82023.

Code availability

SAUCIE is written in Python using the TensorFlow library for deep learning. The source code is available at https://github.com/KrishnaswamyLab/SAUCIE/.

References

37. Moon, K. R. et al. Manifold learning-based methods for analyzing single-cell RNA-sequencing data.*Curr. Opin. Syst. Biol.* **7**, 36–46 (2017).
38. Montufar, G. F., Pascanu, R., Cho, K. & Bengio, Y. On the number of linear regions of deep neural networks. In *Advances in Neural Information Processing Systems Conference 2014* 2924–2932 (JMLR, 2014).
39. Anand, K., Bianconi, G. & Severini, S. Shannon and von Neumann entropy of random networks with heterogeneous expected degree. *Phys. Rev. E* **83**, 036109 (2011).
40. Rellahan, B. L., Bluestone, J. A., Houlden, B. A., Cotterman, M. M. & Matis, L. A. Junctional sequences influence the specificity of gamma/delta T cell receptors. *J. Exp. Med.* **173**, 503–506 (1991).
41. Büttner, M. et al. A test metric for assessing single-cell RNA-seq batch correction. *Nat. Methods* **16**, 43–49 (2019).
42. Luo, B. & Hancock, E. R. Iterative procrustes alignment with the EM algorithm. *Image Vis. Comput.* **20**, 377–396 (2002).
43. Lui, K., Ding, G. W., Huang, R. & McCann, R. Dimensionality reduction has quantifiable imperfections: two geometric bounds. In *Advances in Neural Information Processing Systems 31 (NIPS 2018)* (eds Bengio, S. et al.) 8461–8471 (JMLR, 2018).
44. Tsherniak, A. et al. Defining a cancer dependency map. *Cell* **170**, 564–576 (2017).
45. Yao, Y. et al. The natural killer cell response to West Nile virus in young and old individuals with or without a prior history of infection. *PLoS ONE* **12**, e0172625 (2017).
46. Yao, Y. et al. CyTOF supports efficient detection of immune cell subsets from small samples. *J. Immunol. Methods* **415**, 1–5 (2014).

nature research

Corresponding author(s): Smita Krishnaswamy

Last updated by author(s): Apr 30, 2019

Reporting Summary

Nature Research wishes to improve the reproducibility of the work that we publish. This form provides structure for consistency and transparency in reporting. For further information on Nature Research policies, see Authors & Referees and the Editorial Policy Checklist.

Statistics

For all statistical analyses, confirm that the following items are present in the figure legend, table legend, main text, or Methods section.

n/a	Confirmed	
☐	☒	The exact sample size (*n*) for each experimental group/condition, given as a discrete number and unit of measurement
☒	☐	A statement on whether measurements were taken from distinct samples or whether the same sample was measured repeatedly
☒	☐	The statistical test(s) used AND whether they are one- or two-sided *Only common tests should be described solely by name; describe more complex techniques in the Methods section.*
☒	☐	A description of all covariates tested
☐	☒	A description of any assumptions or corrections, such as tests of normality and adjustment for multiple comparisons
☒	☐	A full description of the statistical parameters including central tendency (e.g. means) or other basic estimates (e.g. regression coefficient) AND variation (e.g. standard deviation) or associated estimates of uncertainty (e.g. confidence intervals)
☒	☐	For null hypothesis testing, the test statistic (e.g. *F*, *t*, *r*) with confidence intervals, effect sizes, degrees of freedom and *P* value noted *Give P values as exact values whenever suitable.*
☒	☐	For Bayesian analysis, information on the choice of priors and Markov chain Monte Carlo settings
☒	☐	For hierarchical and complex designs, identification of the appropriate level for tests and full reporting of outcomes
☒	☐	Estimates of effect sizes (e.g. Cohen's *d*, Pearson's *r*), indicating how they were calculated

Our web collection on statistics for biologists contains articles on many of the points above.

Software and code

Policy information about availability of computer code

Data collection	All code was written in Python and the link to software is in the manuscript and available at: https://github.com/KrishnaswamyLab/SAUCIE
Data analysis	The link to the Python code is given in the manuscript and available at https://github.com/KrishnaswamyLab/SAUCIE

For manuscripts utilizing custom algorithms or software that are central to the research but not yet described in published literature, software must be made available to editors/reviewers. We strongly encourage code deposition in a community repository (e.g. GitHub). See the Nature Research guidelines for submitting code & software for further information.

Data

Policy information about availability of data

All manuscripts must include a data availability statement. This statement should provide the following information, where applicable:

- Accession codes, unique identifiers, or web links for publicly available datasets
- A list of figures that have associated raw data
- A description of any restrictions on data availability

Proper references to public data are all given in the manuscript.

Field-specific reporting

Please select the one below that is the best fit for your research. If you are not sure, read the appropriate sections before making your selection.

☒ Life sciences ☐ Behavioural & social sciences ☐ Ecological, evolutionary & environmental sciences

Life sciences study design

All studies must disclose on these points even when the disclosure is negative.

Sample size	*Describe how sample size was determined, detailing any statistical methods used to predetermine sample size OR if no sample-size calculation was performed, describe how sample sizes were chosen and provide a rationale for why these sample sizes are sufficient.*
Data exclusions	*Describe any data exclusions. If no data were excluded from the analyses, state so OR if data were excluded, describe the exclusions and the rationale behind them, indicating whether exclusion criteria were pre-established.*
Replication	*Describe the measures taken to verify the reproducibility of the experimental findings. If all attempts at replication were successful, confirm this OR if there are any findings that were not replicated or cannot be reproduced, note this and describe why.*
Randomization	*Describe how samples/organisms/participants were allocated into experimental groups. If allocation was not random, describe how covariates were controlled OR if this is not relevant to your study, explain why.*
Blinding	*Describe whether the investigators were blinded to group allocation during data collection and/or analysis. If blinding was not possible, describe why OR explain why blinding was not relevant to your study.*

Reporting for specific materials, systems and methods

We require information from authors about some types of materials, experimental systems and methods used in many studies. Here, indicate whether each material, system or method listed is relevant to your study. If you are not sure if a list item applies to your research, read the appropriate section before selecting a response.

Materials & experimental systems

n/a	Involved in the study
☐	☐ Antibodies
☐	☐ Eukaryotic cell lines
☐	☐ Palaeontology
☐	☐ Animals and other organisms
☐	☐ Human research participants
☐	☐ Clinical data

Methods

n/a	Involved in the study
☐	☐ ChIP-seq
☐	☐ Flow cytometry
☐	☐ MRI-based neuroimaging

Antibodies

Antibodies used	*Describe all antibodies used in the study; as applicable, provide supplier name, catalog number, clone name, and lot number.*
Validation	*Describe the validation of each primary antibody for the species and application, noting any validation statements on the manufacturer's website, relevant citations, antibody profiles in online databases, or data provided in the manuscript.*

Eukaryotic cell lines

Policy information about cell lines

Cell line source(s)	*State the source of each cell line used.*
Authentication	*Describe the authentication procedures for each cell line used OR declare that none of the cell lines used were authenticated.*
Mycoplasma contamination	*Confirm that all cell lines tested negative for mycoplasma contamination OR describe the results of the testing for mycoplasma contamination OR declare that the cell lines were not tested for mycoplasma contamination.*
Commonly misidentified lines (See ICLAC register)	*Name any commonly misidentified cell lines used in the study and provide a rationale for their use.*

Palaeontology

Specimen provenance	*Provide provenance information for specimens and describe permits that were obtained for the work (including the name of the issuing authority, the date of issue, and any identifying information).*
Specimen deposition	*Indicate where the specimens have been deposited to permit free access by other researchers.*
Dating methods	*If new dates are provided, describe how they were obtained (e.g. collection, storage, sample pretreatment and measurement),*

Dating methods

where they were obtained (i.e. lab name), the calibration program and the protocol for quality assurance OR state that no new dates are provided.

☐ Tick this box to confirm that the raw and calibrated dates are available in the paper or in Supplementary Information.

Animals and other organisms

Policy information about studies involving animals; ARRIVE guidelines recommended for reporting animal research

Laboratory animals

For laboratory animals, report species, strain, sex and age OR state that the study did not involve laboratory animals.

Wild animals

Provide details on animals observed in or captured in the field; report species, sex and age where possible. Describe how animals were caught and transported and what happened to captive animals after the study (if killed, explain why and describe method; if released, say where and when) OR state that the study did not involve wild animals.

Field-collected samples

For laboratory work with field-collected samples, describe all relevant parameters such as housing, maintenance, temperature, photoperiod and end-of-experiment protocol OR state that the study did not involve samples collected from the field.

Ethics oversight

Identify the organization(s) that approved or provided guidance on the study protocol, OR state that no ethical approval or guidance was required and explain why not.

Note that full information on the approval of the study protocol must also be provided in the manuscript.

Human research participants

Policy information about studies involving human research participants

Population characteristics

Describe the covariate-relevant population characteristics of the human research participants (e.g. age, gender, genotypic information, past and current diagnosis and treatment categories). If you filled out the behavioural & social sciences study design questions and have nothing to add here, write "See above."

Recruitment

Describe how participants were recruited. Outline any potential self-selection bias or other biases that may be present and how these are likely to impact results.

Ethics oversight

Identify the organization(s) that approved the study protocol.

Note that full information on the approval of the study protocol must also be provided in the manuscript.

Clinical data

Policy information about clinical studies

All manuscripts should comply with the ICMJE guidelines for publication of clinical research and a completed CONSORT checklist must be included with all submissions.

Clinical trial registration

Provide the trial registration number from ClinicalTrials.gov or an equivalent agency.

Study protocol

Note where the full trial protocol can be accessed OR if not available, explain why.

Data collection

Describe the settings and locales of data collection, noting the time periods of recruitment and data collection.

Outcomes

Describe how you pre-defined primary and secondary outcome measures and how you assessed these measures.

ChIP-seq

Data deposition

☐ Confirm that both raw and final processed data have been deposited in a public database such as GEO.

☐ Confirm that you have deposited or provided access to graph files (e.g. BED files) for the called peaks.

Data access links
May remain private before publication.

For "Initial submission" or "Revised version" documents, provide reviewer access links. For your "Final submission" document, provide a link to the deposited data.

Files in database submission

Provide a list of all files available in the database submission.

Genome browser session
(e.g. UCSC)

Provide a link to an anonymized genome browser session for "Initial submission" and "Revised version" documents only, to enable peer review. Write "no longer applicable" for "Final submission" documents.

Methodology

Replicates

Describe the experimental replicates, specifying number, type and replicate agreement.

Sequencing depth	*Describe the sequencing depth for each experiment, providing the total number of reads, uniquely mapped reads, length of reads and whether they were paired- or single-end.*
Antibodies	*Describe the antibodies used for the ChIP-seq experiments; as applicable, provide supplier name, catalog number, clone name, and lot number.*
Peak calling parameters	*Specify the command line program and parameters used for read mapping and peak calling, including the ChIP, control and index files used.*
Data quality	*Describe the methods used to ensure data quality in full detail, including how many peaks are at FDR 5% and above 5-fold enrichment.*
Software	*Describe the software used to collect and analyze the ChIP-seq data. For custom code that has been deposited into a community repository, provide accession details.*

Flow Cytometry

Plots

Confirm that:

☐ The axis labels state the marker and fluorochrome used (e.g. CD4-FITC).

☐ The axis scales are clearly visible. Include numbers along axes only for bottom left plot of group (a 'group' is an analysis of identical markers).

☐ All plots are contour plots with outliers or pseudocolor plots.

☐ A numerical value for number of cells or percentage (with statistics) is provided.

Methodology

Sample preparation	*Describe the sample preparation, detailing the biological source of the cells and any tissue processing steps used.*
Instrument	*Identify the instrument used for data collection, specifying make and model number.*
Software	*Describe the software used to collect and analyze the flow cytometry data. For custom code that has been deposited into a community repository, provide accession details.*
Cell population abundance	*Describe the abundance of the relevant cell populations within post-sort fractions, providing details on the purity of the samples and how it was determined.*
Gating strategy	*Describe the gating strategy used for all relevant experiments, specifying the preliminary FSC/SSC gates of the starting cell population, indicating where boundaries between "positive" and "negative" staining cell populations are defined.*

☐ Tick this box to confirm that a figure exemplifying the gating strategy is provided in the Supplementary Information.

Magnetic resonance imaging

Experimental design

Design type	*Indicate task or resting state; event-related or block design.*
Design specifications	*Specify the number of blocks, trials or experimental units per session and/or subject, and specify the length of each trial or block (if trials are blocked) and interval between trials.*
Behavioral performance measures	*State number and/or type of variables recorded (e.g. correct button press, response time) and what statistics were used to establish that the subjects were performing the task as expected (e.g. mean, range, and/or standard deviation across subjects).*

Acquisition

Imaging type(s)	*Specify: functional, structural, diffusion, perfusion.*
Field strength	*Specify in Tesla*
Sequence & imaging parameters	*Specify the pulse sequence type (gradient echo, spin echo, etc.), imaging type (EPI, spiral, etc.), field of view, matrix size, slice thickness, orientation and TE/TR/flip angle.*
Area of acquisition	*State whether a whole brain scan was used OR define the area of acquisition, describing how the region was determined.*

Diffusion MRI ☐ Used ☐ Not used

Preprocessing

Preprocessing software

Provide detail on software version and revision number and on specific parameters (model/functions, brain extraction, segmentation, smoothing kernel size, etc.).

Normalization

If data were normalized/standardized, describe the approach(es): specify linear or non-linear and define image types used for transformation OR indicate that data were not normalized and explain rationale for lack of normalization.

Normalization template

Describe the template used for normalization/transformation, specifying subject space or group standardized space (e.g. original Talairach, MNI305, ICBM152) OR indicate that the data were not normalized.

Noise and artifact removal

Describe your procedure(s) for artifact and structured noise removal, specifying motion parameters, tissue signals and physiological signals (heart rate, respiration).

Volume censoring

Define your software and/or method and criteria for volume censoring, and state the extent of such censoring.

Statistical modeling & inference

Model type and settings

Specify type (mass univariate, multivariate, RSA, predictive, etc.) and describe essential details of the model at the first and second levels (e.g. fixed, random or mixed effects; drift or auto-correlation).

Effect(s) tested

Define precise effect in terms of the task or stimulus conditions instead of psychological concepts and indicate whether ANOVA or factorial designs were used.

Specify type of analysis: ☐ Whole brain ☐ ROI-based ☐ Both

Statistic type for inference
(See Eklund et al. 2016)

Specify voxel-wise or cluster-wise and report all relevant parameters for cluster-wise methods.

Correction

Describe the type of correction and how it is obtained for multiple comparisons (e.g. FWE, FDR, permutation or Monte Carlo).

Models & analysis

n/a	Involved in the study
☐	☐ Functional and/or effective connectivity
☐	☐ Graph analysis
☐	☐ Multivariate modeling or predictive analysis

Functional and/or effective connectivity

Report the measures of dependence used and the model details (e.g. Pearson correlation, partial correlation, mutual information).

Graph analysis

Report the dependent variable and connectivity measure, specifying weighted graph or binarized graph, subject- or group-level, and the global and/or node summaries used (e.g. clustering coefficient, efficiency, etc.).

Multivariate modeling and predictive analysis

Specify independent variables, features extraction and dimension reduction, model, training and evaluation metrics.