Fast Optimization of Robust Transcriptomics Embeddings using Probabilistic Inference Autoencoder Networks for multi-Omics

Ning Wang[1,3]*, David Turner[1], Hannah Feinberg[1], Victor Eduardo Nieto Caballero[1], Dan Yuan[3], Nathaniel Scott[1], Christopher Cardenas[1], Michael DeBerardine[1], Shu Dan[1], Lakme Caceres[1], Jessica Schembri[1], Zizhen Yao[3], Changkyu Lee[3], Jonathan W. Pillow[1,4,5], Fenna M. Krienen[1,2]*

* Corresponding author
1. Princeton Neuroscience Institute (PNI)
2. Princeton Program in Quantitative and Computational Biology (QCB)
3. Allen Institute for Brain Science (AIBS)
4. Princeton Center for Statistics and Machine Learning (CSML)
5. Princeton Program in Applied and Computational Mathematics (PACM)

SUMMARY

Advances in single-cell genomics technologies enable the routine acquisition of atlases with millions of cells. These datasets often include multiple sources of variation, such as donors, sequencing platforms, developmental timepoints, and species. Although these covariates provide new opportunities for discovery, they also present challenges for downstream analyses. To mitigate unwanted sources of variation, dataset integration is the starting point for most analyses. However, existing methods struggle with integrating large and complex datasets. To address these limitations, we developed PIANO, a variational autoencoder framework that uses a negative binomial generalized linear model for stronger batch correction, and code compilation for up to ten times faster training than existing tools. We first demonstrate performant integration compared to commonly used integration methods on single-species datasets. We then show PIANO enables superior analyses of multiple atlases, solving challenging integration tasks across sequencing platforms, developmental timepoints, and species, while simultaneously preserving desired biological signals. Our contributions include a novel, high-performance integration method and recommendations for integration applications.

KEYWORDS

- Single cell RNA sequencing, Multi-omics, Integration, Cross species atlases, Transcriptomics, Spatial Transcriptomics, Deep Learning, Methods Development

INTRODUCTION

Single cell sequencing enables the creation of cross-species and cross-modality atlases with millions of cells and nuclei, such as whole mouse and human brain atlases[1,2]. This powers large consortium-based efforts to map the cellular composition of entire organs or even organisms, such as those from the BRAIN Initiative Cell Atlas Network (BICAN), Human Cell Atlas (HCA), and Chan Zuckerberg Initiative (CZI). Comparing data from these large studies

enables exploration of the evolutionary conservation and divergence of cell types across multiple species, developmental timepoints, and modalities[1–4]. Consequently, such atlases now scale to tens of millions of cells, which calls for powerful new tools for analyzing the data.

While combining many datasets enables novel biological insights, single cell sequencing datasets are still heavily impacted by data sparsity and multiple sources of variation. Sources of variation such as the tissue donor and sequencing platform – often referred to collectively as batch effects – confound biological discoveries and make it difficult to compare cell types across experimental conditions. Therefore, one goal of data integration is to find a more informative, low-dimensional representation of the original data. Ideally, the integrated latent space representation should remove batch effects while preserving relevant biological information (e.g. cell types).

To enable robust comparisons across datasets, multiple methods have been developed for projecting single cell or single nucleus sequencing data to a shared latent space embedding[5]. One commonly used approach is Seurat's canonical correlation analysis (CCA), which finds anchors between batches by identifying pairs of similar cells across datasets[6]. However, using pairwise anchors may over-mix data in situations with imbalanced cell type distributions between pairs. While Seurat may be sufficient for simpler tasks, pairwise anchor calculations scale quadratically with the number of batches[6,7]. Moreover, the sparse matrices used in Seurat's implementation cannot handle more than 2^{31} - 1 non-zero gene counts in memory, which further limits its scalability. Another popular method, Harmony, uses expectation-maximization (EM) to perform batch correction[5]. However, recent benchmarking studies suggest Harmony also struggles on complex integration tasks[7].

An alternative solution is to use a variational autoencoder (VAE)[8] for integration. A VAE is a powerful, non-linear, generative model that uses an encoder-decoder architecture to explain high-dimensional data using a low-dimensional latent variable. Through dimensionality reduction, the VAE extracts the most meaningful information from the data, such as information about cell type clusters. The generative component of the model, or decoder, describes a transformation from this low-dimensional latent space to the high-dimensional space of the raw counts data. The other component of a VAE is the recognition model, or encoder, which enables inference of the latent variables from an observed single cell dataset. When used for integration, the VAE trains on data from multiple batches and integrates the data by performing batch correction, minimizing the differences between batches.

One popular VAE method, scVI, integrates single cell datasets using a negative binomial (NB) distribution to account for over-dispersion in the raw gene counts[9]; scVI performs batch correction by conditioning the over-dispersion term for each gene on the primary source of variation[9]. Although the scVI decoder can take additional covariates as input, each NB distribution is only conditioned on one primary batch key. Moreover, despite benefits from graphics processing unit (GPU) acceleration as an end-to-end trained neural network, scVI's training time scales linearly with the size of the data[7,9]. As a result, scVI may take days to train on large datasets with millions of cells, which delays downstream analyses.

To address these limitations, we introduce PIANO: Probabilistic Inference Autoencoder Networks for multi-Omics, a novel variational autoencoder framework for inferring integrated latent space representations for single cell transcriptomics data. We utilize a generalized linear model (GLM) to fit a negative binomial (NB) distribution for each gene to address unwanted

sources of variation. Rather than relying on a primary batch key like current VAE methods[9], PIANO accounts for all covariates using its NB-GLM. PIANO integration achieves superior or comparable biological conservation, which measures preservation of cell type clusters, with stronger batch correction, which measures mixing of data across batches, on multiple integration tasks, as measured by commonly used integration benchmarks. Moreover, our light-weight code implementation that utilizes code compilation leads to substantially faster training speed, resulting in up to 10 times faster training than current VAE models.

RESULTS

NB-GLM variational autoencoder enables integration of gene count data

To enable integration by reducing batch effects, PIANO learns a latent representation of the data using a variational autoencoder (VAE)[8] that incorporates external covariates (Figure 1). PIANO uses a deep neural network (DNN) inference model, or "encoder", to perform a non-linear mapping from vectors of gene-count observations to lower-dimensional latent variables, learning an approximate distribution over the latent representation for each cell (Figure 1). To describe gene count data, PIANO uses a generative model, or “decoder”, which uses another DNN to perform a non-linear mapping between latent variables and their covariates to the parameters of a negative binomial (NB) distribution that models the original observations. Due to the information bottleneck imposed by low-dimensionality, the latent variable must learn the most informative representation of the data to help the decoder model the original data. The final layer of the decoder incorporates a NB-GLM — a generalized linear model (GLM) with NB observations for each gene, which combines its DNN outputs with linear weights on the library size (i.e., total counts per cell) and batch-related covariates, such as donor or species (see Methods, Supplemental Table 1). Since these covariates are explicitly provided to both the decoder and to the GLM, the latent variable itself does not need to inform the decoder about covariates for modeling the data and instead encodes cell-type information with greatly reduced batch effects. This GLM framework accounts for multiple overlapping sources of variation, which offers greater flexibility and stronger batch correction than scVI's conditional batch key (see Methods).

PIANO integrates across single-species donor and technical variability

We first demonstrated PIANO integration on five single-species cases. Datasets varied in the number of cells (16k-373k), cell types (11-81) and tissues, but all contained known batch effects, such as donors and sequencing platforms[10–21] (Figure 2). To quantify integration performance, we used two metrics for biological conservation: normalized mutual information (NMI) and adjusted Rand index (ARI), and two metrics for batch correction: K-nearest-neighbors batch effect test (KBET) and integration local inverse Simpson's index (iLISI) (Methods). PIANO has a strong overall integration performance compared with other integration methods (Seurat, scVI, and Harmony) and an unintegrated baseline using principal component analysis (PCA) (Figure 2a, Supplemental Table 2). Due to stochasticity in model initialization and the relatively small dataset sizes, both the PIANO and scVI integrations were repeated five times, and the integration benchmarking metrics were averaged over the five integrations. For average

biological conservation between NMI and ARI, PIANO had comparable performance to scVI and generally stronger performance to Seurat (Figure 2b, Supplemental Table 3). For average batch correction between KBET and iLISI, Seurat generally had the strongest performance, followed by PIANO (Figure 2c, Supplemental Table 4). However, Seurat may have overcorrected for batch effects, which may have contributed to Seurat's underperformance in biological conservation (Figure 2b). While Seurat tended to overintegrate, Harmony and PCA had generally lower integration performance for most datasets across all metrics (Figure 2a-c). This can be appreciated from the uniform manifold approximation and projection (UMAP) visualizations of the respective latent spaces, where over-integrated data results in blended cell type clusters (Supplemental Figures 1-5). In contrast, Harmony tends to underintegrate the data (Supplemental Figures 1-5). Individual integration UMAPs and benchmarking scores can be found in supplemental figures and tables (Supplemental Figures 1-5, Supplemental Tables 5-9).

For the smaller datasets, scVI and Seurat had similar runtimes, but on the larger heart dataset, Seurat took over twice as long to run as scVI (Figure 2d). In contrast, PIANO was faster to train than all integration methods besides Harmony for every dataset; PIANO was still faster even when including compilation time, which accounted for the majority of the total training time on the smaller datasets (Figure 2d). On the larger heart atlas (n=343.5k cells), where compilation time accounts for a smaller proportion of training time, PIANO using one worker in its GPU memory mode (Methods) was seven times faster than scVI using eleven workers and 15.5x faster than Seurat (Figure 2d); both PIANO and scVI models were trained using the same type of hardware: one NVIDIA A100 GPU with 40Gb of GPU memory. Runtimes are in Supplemental Table 10.

PIANO integrates atlas-level data across the primate basal ganglia

To better understand the behavior of VAE models for single cell integration, we investigated multiple variations of feature selection and model design using a large multi-species brain cell atlas of the basal ganglia (Figure 3), acquired as part of the BRAIN Initiative Cell Atlas Network (BICAN) (Johansen et al., 2025). The Human and Mammalian Brain Cell Atlas (HMBA) contains RNA sequencing data from nearly 2 million single nuclei sampled across human, macaque, and marmoset basal ganglia structures. Each species' data were originally computationally integrated using scVI to generate a consensus cell taxonomy at two hierarchical levels: cell Class (12 Classes) and cell Group (61 Groups) (Johansen et al., 2025).

To balance the number of cells between species, we subset the data to include up to 6,400 cells per Group for each species. Using the Group labels to assess biological conservation, PIANO performed a better separation of cell types and mixing between donors and species (Figure 3a-c) compared to scVI (Figure 3 d-f). We found that a model architecture with 3 layers, 256 hidden nodes, and 32 latent dimensions had a well-balanced integration performance (Figure 3g, Supplemental Figure 6). As the number of training epochs increased, the batch correction improved while the biological conservation remained stable (Figure 3g). In contrast, scVI had a less noticeable improvement in integration performance over training (Figure 3g). While PIANO had similar biological conservation of Groups to scVI, PIANO had substantially higher batch correction across species (Figure 3g). Additionally, both PIANO and scVI outperformed using a PCA latent space, which offered no batch correction (Figure 3g).

Gene selection is an important parameter for integration tasks, particularly when integrating datasets across species. We first subset the data to 1:1 homologous genes[22] and then selected the top highly variable genes (HVGs) across species for integration using the "Seurat V3" flavor for highly variable gene selection, using species as the "batch_key". We performed a hyperparameter sweep over the number of top HVGs and found that using 4096 HVGs for integrating the HMBA datasets had robust performance (Figure 3h). Surprisingly, the same number of randomly selected genes had comparable integration performance with similar batch correction and slightly worse biological conservation (Figure 3i). This suggests that PIANO can robustly integrate data given sufficient high-quality, gene expression training data.

Several atlas studies have prioritized functional gene sets to construct single-species cell type taxonomies. The Allen Institute whole mouse brain (WMB) atlas compared cross-validation accuracy of clustering using all cluster-wise differentially expressed genes (n=8,460 genes) against several functional gene sets including transcription factors (TFs, n=534 genes), cell adhesion molecules (n=857 genes) and other functional classes (n=541 genes)[2]. This analysis demonstrated that cross-validation accuracy using TFs was comparable to all expressed genes for discriminating between 338 subclasses of cells[2]. In the WMB taxonomy, the Subclass is considered an intermediate level of the atlas[2] and approximates the Group level in the HMBA dataset (Johansen et al., 2025). We examined whether functional gene classes performed similarly to large sets of highly variable genes for cross-species integration (Figure 3j). The larger functional gene sets, such as the orthologous synapse related genes from the UniprotKB[23] (838 genes) and orthologous transcription factors from AnimalTFDB4[24] (1,119 genes) showed similar performance to the 4096 highly variable genes and outperformed the orthologous extracellular matrix genes (414 genes) (Figure 3j). This suggests that functional gene sets that are sufficiently large contain enough information to capture the nuances between cell type clusters (Figure 3d-g). UMAPs for PCA can be found in Supplemental Figure 7, and specific benchmarking scores can be found in Supplemental Tables Supplemental Table 11-13.

One mode of evolutionary divergence could come as complete gain or loss of a cell type[25]. As such, an ideal cross-species integration should preserve species-specific cell type populations[26]. We performed two *in silico* perturbation experiments on evolutionarily conserved cell types to determine whether or not PIANO and scVI would overcorrect for species effects. To determine if a species-specific cell type would be merged into a related cell type shared with other species, we first withheld only the hybrid medium spiny neurons (Hybrid MSNs) Subclass from the non-human primate datasets, such that only the human basal ganglia dataset retained them. Both PIANO and scVI kept these Hybrid MSNs separate from the related D1 and D2 MSNs (Supplemental Figure 8). We then withheld only the non-human primate D1 MSNs. Since the D1 MSNs are much more transcriptionally similar to the D2 MSNs than are the Hybrid MSNs, this constitutes a more difficult perturbation. Initially, both PIANO and scVI mixed the D1 and D2 MSNs, resulting in overlapping cell type clusters (Supplemental Figure 9); however, by lowering the value of the beta hyperparameter while training PIANO (see Methods), the D1 MSNs were re-separated from the D2 MSNs (Supplemental Figure 9). These results suggest PIANO may require tuning to better retain dataset-specific populations, depending on the relative similarity between shared and imbalanced cell populations across datasets.

Integration is still an unsolved problem for datasets with large noise effects

Cross-species integrations become more challenging with increased evolutionary divergence, particularly when each species' dataset also contains major within-dataset covariates such as different developmental timepoints. To assess a more complex scenario we evaluated PIANO integration performance between the mouse cell atlas (MCA)[27] and human cell landscape (HCL) datasets[28] (Figure 4). Not only did these datasets have many cell types across species and developmental time points, they also had low signal-to-noise, with fewer counts captured per cell compared to more recently collected datasets[26] due to using an older Microwell-Seq platform[27,28]. Consequently, we found it difficult to maintain separable cell type clusters while mixing across species for PIANO (Figure 4a-b) and scVI (Figure 4c-d, Figure 4 S1), which resulted in lower overall integration performance than previous integration examples. PIANO and scVI had similar biological conservation scores, and scVI performed very little batch correction (Figure 4e). Moreover, we found that PIANO trained up to ten times faster than scVI as the number of cells increased (Figure 4f). UMAPs for PCA (including the cell types legend) can be found in Supplemental Figure 10, and benchmarking metrics and runtimes can be found in supplementary tables (Supplemental Table 14-15).

PIANO outperforms scVI at mammalian whole brain integration

Another major challenge was to integrate a recently published whole human brain atlas[1] with the whole mouse brain atlas[2]. Due to computational runtime limitations of scVI, we subset the data to up to 100 cells per highest resolution cluster in each atlas. Currently, there does not yet exist a well-established, cross-species consensus taxonomy between human and mouse whole brain cell types; developing a cross-species whole brain atlas is beyond the scope of this manuscript. Instead, we compared the biological conservation of brain regions and found cleanly separated clusters in the latent space embeddings (Figure 5a). Additionally, we found a good mixture of cells between human and mouse species across brain regions (Figure 5b). We also observed that the PIANO integration had comparable region separation to the scVI integration, and the PIANO batch correction performance steadily improved as the number of training epochs increased (Figure 5c). scVI regions were less separated (Figure 5d) and less mixed across species (Figure 5e). Although the PIANO integration benchmarking metric scores are lower than the previous examples, this is likely due to the complexity of the data and a limitation of using coarse brain regions as the "cell type" annotation. Moreover, the data, while not fully mixed across species like in the basal ganglia integration (Figure 3), is still fairly close together. The UMAPs for PCA can be found in Supplemental Figure 11, and benchmarking metrics scores can be found in Supplemental Table 16.

PIANO enables multi-modal integration with spatial transcriptomics

Integration across data modalities empowers biological discovery. Spatial transcriptomics technologies, such as multiplexed error-robust fluorescence in situ hybridization (MERFISH)[29], enable identifying exact physical locations of cells in a slice of tissue, which is not known in single cell RNA transcriptomics data; however, this spatial resolution comes at the cost of capturing fewer genes than single cell data, which poses a new challenge for integration. The Broad Institute created a whole mouse brain spatial atlas containing a complete spatial

transcriptomics series of four mouse brains[30]; due to having more genes, we decided to use the Broad Institute's MERFISH data with 1,122 genes shared across panels[30] instead of the Allen Institute's MERFISH data with a 500 gene panel[2] for demonstrating spatial integration.

To obtain cell type labels, the Broad Institute spatial atlas was originally integrated with the Allen Institute's whole mouse brain single cell atlas[2] using Seurat (CCA)[30,31]. Given PIANO and scVI's superior performance to Seurat for data integration, we revisited integrating the Broad Institute's whole mouse brain spatial transcriptomics atlas with the Allen Institute's whole mouse brain single cell transcriptomics atlas. We first subset the genes in the single cell transcriptomics data and spatial transcriptomics data to the 1,122 genes shared in all datasets[2,30]; we then subset the cells in each datasets to up to 6400 cells per Subclass to obtain a balanced dataset for integration. We found good mixing between single cell and spatial modalities (Figure 6a) and good separation of cell type Classes (Figure 6b). Since the Allen Institute whole mouse brain taxonomy used hierarchical clustering for cell types, we demonstrated an illustrative example of hierarchical reintegration for the cells from Class 13 CNU-HYa Glut at the Subclass level (Figure 6c). Since the training time for PIANO is approximately linear with respect to the number of cells, PIANO can perform hierarchical integration with a linear runtime cost with respect to the number of levels in the hierarchy. This enables efficient and comprehensive analyses of large atlases at multiple cell type resolutions. In contrast, scVI had less mixing across modalities (Figure 6d), less separation of classes (Figure 6e), and less separation of subclasses (Figure 6f). Moreover, PIANO had better mixing of modalities for subclasses (Figure 6g) than scVI (Figure 6h). Again, we observed superior biological conservation and batch correction compared to scVI for both the whole brain integrations (Figure 6i) and Class 13 CNU-HYa Glut integration (Figure 6j). UMAPs for PCA can be found in Supplemental Figure 12, and benchmarking scores can be found in supplemental tables 17-18.

PIANO enables integration across species and developmental timepoints

Developmental datasets pose difficult challenges for integration tasks, as cell type identities are especially dynamic during maturation[26]. This challenge is further exacerbated when attempting to integrate across species, since developmental timepoints may not be aligned. In a recent study, we compared astrocyte development across embryonic and postnatal timepoints in mice and marmosets, but were not able to successfully integrate the species datasets together using existing methods[26].

Using PIANO, we obtained robust integrations of the complete dataset, including neurons and non-neuronal cells[26] (Figure 7). We found strong integration with good mixing across species (Figure 7a), separation of cell types (Figure 7b) and superclusters (Figure 7c). In contrast, scVI had less mixing across species (Figure 7d), and similar separation of cell types (Figure 7e) and superclusters (Figure 7f). Additionally, we found separation along certain developmental time points, such as cells in the embryonic stage distinctly separated from the clusters in the later time points (Figure 7g), which were mixed together in the previous single species integrations[26]. Moreover, we preserved distinct region differences observed in both neurons and glia (Figure 7h), recapitulating region specificity of glial cell types as described in the original atlas manuscript[26]. Although scVI similarly captured these differences (Figure 7i-j), it struggled with mixing data across species (Figure 7k). Furthermore, the original manuscript also

identified two species-specific neuronal superclusters: an embryonic mouse type 'Inh Str Immature', and a marmoset thalamic GABAergic type 'Inh Thal MB-der'; both species-specific cell types are preserved in PIANO's integration (Figure 7a-c,g-h). These findings show that PIANO can perform highly-robust cross-species integration comparisons between evolutionary distant species, while still preserving biological signals attributable to regional and developmental sources. UMAPs for PCA can be found in Supplemental Figure 13, and benchmarking metrics are in Supplemental Table 19.

DISCUSSION

PIANO enables fast and robust integrations

We introduced PIANO, a powerful, scalable, and efficient framework for integrating single cell multi-omics data. We showed that PIANO enables 1) fast integrations that require a small fraction of the runtime cost of existing tools and 2) produces high-quality latent space representations that both preserve desired biological features and integrate data between multiple batches. We demonstrate applications of our integration framework across a wide variety of datasets over multiple species, tissues, sequencing platforms, and developmental timepoints. The broader impacts of our method include greatly expanding the accessibility of powerful GPU-based integration tools by reducing the runtime cost by an order of magnitude while producing superior or comparable integration results. We also dramatically speed up training with our light-weight codebase, enabling atlas-scale integrations in less than an hour.

Model parameter selection

The default PIANO model architecture with 3 hidden layers, 256 hidden nodes, and 32 latent dimensions worked well for all integrations with 200 training epochs. The most important PIANO hyperparameter to adjust is the max beta hyperparameter, where a higher max value for beta results in stronger batch correction in exchange for some biological conservation of cell types. For most of our integrations, we used our default max value of 1.0 after 400 training epochs (see Methods). In situations where the default PIANO beta value results in over-mixing across batches, we recommend reducing the max value. This is exemplified by the withheld D1 MSN primate basal ganglia integration, where reducing the max value from 1.0 to 0.5 recaptured the human-specific perturbed cell type (Figure 3 S2.2). PIANO trains quickly using code compilation and GPU acceleration, which makes hyperparameter sweeps computationally tractable. We also recommend first identifying cell types within each species before performing cross species integration to avoid missing species-specific signals.

Gene selection:

Previous studies have found that some functional gene sets, such as transcription factor genes, are able to more accurately predict cell types than other gene sets, such as cell adhesion molecule encoding genes[2]. We found that the functional gene sets were similarly able to integrate data across species, where gene sets with larger numbers of genes generally had better performance (Figure 3j). This is consistent with our comparison over the number of highly variable genes, where more genes used resulted in slightly higher biological conservation in

exchange for small decreases in batch correction (Figure 3h), and our comparison with increasing numbers of random genes shows a similar pattern (Figure 3i). Although functional gene sets may be informative, the number of genes used for integration is also an important consideration.

Data quality influences integration results

One notable observation is the variation in the quality of integrations between various atlases. In particular, the mouse and marmoset developmental integration from Schroeder et al.[26] produced very clean integrations with both exceptionally high biological conservation and batch correction scores (Figure 6), whereas the mouse and human cross-tissue integration between the MCA and HCL datasets had far lower integration performances (Figure 4). While this may be impacted by sampling different tissue types, there is also a stark difference in the sequencing depth and ultimate UMI yield between the datasets. Schroeder et al. targeted 40,000 sense reads per nucleus[26], whereas the HCL obtained a substantially fewer 3000 reads per cell[28]. Moreover, Schroeder et al. used the 10x 3' v3 chemistry[26], whereas the HCL used a Microwell-seq platform[28]. As a result, the mean number of unique molecular identifiers (UMIs) per nuclei in each batch had a mean of 6,085 UMIs per nuclei across 144 batches in Schroeder et al.[26], whereas the HCL data had a mean of less than 934 UMIs per cell in each tissue across 112 tissue samples[28]. For the whole brain integrations, the Allen Institute's whole mouse brain atlas achieved an average of 54,379 reads per cell for the 10Xv2 data and 83,190 reads per cell for the 10Xv3 data[2], and the whole human brain atlas achieved approximately 100,000 reads per nuclei[1]. Note these datasets count both sense and anti-sense reads, while the Schroeder et al. study only counted sense reads. As anti-sense reads can make up to 40% of reads, these datasets are all generally comparable in sequencing depth.

Dramatic differences in dataset depth can contribute to the large discrepancies in the quality of integration results. This reinforces the crucial need for high-quality sequencing data to distinguish between evolutionary divergences across species with confounding noise artifacts. Just as Seurat was inspired by pointillism painting, the name of the our method, PIANO, serves as a metaphor between integrating complex atlases to performing piano concertos, such as Rachmaninoff's Piano Concerto No. 2 in C minor[32,33]. Just as the quality of the performance depends on individual contributions of the pianist, conductor, and orchestra, the success of cross-species integrations relies on having large quantities of high-quality data. While the PIANO integrations increased batch correction across epochs at a higher rate than scVI across most datasets, this performance improvement was more apparent in datasets in which a large number of cells were available for each cell type when training the model (Methods). This further highlights the importance of high-quality training data for robust integration performance.

Limitations

One limitation of our model is that it relies on modern GPU acceleration to make the most of its performance improvements. While we tested our model for both x86 and ARM CPU architectures, the best performance improvements come with its compiled mode, which requires the use of a GPU that can support torch.compile optimizations. Moreover, these compilations introduce up to two minutes of additional compilation time overhead costs, which make up a larger proportion of runtime for smaller datasets. These hardware requirements can be

mitigated using institutional high-performance computing clusters (HPCs) or cloud computing platforms.

Finally, as with most machine learning methods, the quality of the results depends on the quality of the input data. Using more high-quality data not only results in more substantial integration and runtime performance improvements compared to other integration methods, but also further enriches the training of the model. Moreover, with single cell integration, there still exists a fundamental tradeoff between batch correction and biological conservation[34].

Related work

The NB-GLM modeling used in the VAE decoder was inspired by the NB-GLM used by scTransform for normalizing scRNAseq data[35]. Other commonly used integration tools include Seurat (CCA)[6], Harmony[5], scVI[9], and others. Additionally, some integration methods attempt to use a semi-supervised approach, such as scANVI[36]. However, these methods rely on the robustness of the assigned cell type labels and may bias results against novel biology or rarer cell types, and do not model relationships between similar cell types. Moreover, using a semi-supervised approach may introduce training instability and longer runtimes.

Acknowledgements

This publication was supported by and coordinated through the Brain Initiative Cell Atlas Network (BICAN). This work was supported by National Institutes of Health UM1MH130981, DP2MH140136, and the Klingenstein–Simons Fellowship (F.M.K.). N.W. was supported by The McDonnell Fellows in Neuroscience at Princeton University and National Institutes of Health T32MH065214. This work was completed in part at the Princeton Open Hackathon, part of the Open Hackathons program. The authors would like to acknowledge OpenACC-Standard.org for their support. The authors would like to acknowledge the Allen Institute for Neural Dynamics for their computational support with Code Ocean. We wish to thank the Allen Institute founder, Paul G. Allen, for his vision, encouragement, and support.

METHODS

PIANO model architecture

PIANO uses approximate Bayesian inference to train a modified variational autoencoder (VAE)[8] using an encoder-decoder architecture. The encoder, or recognition model, assumes an isotropic Gaussian prior and learns a probabilistic, posterior latent space representation for the observed input data X (Figure 1). This latent space uses an independent mean and standard deviation for each dimension (Figure 1), which enables sampling via a reparameterization trick[8]. The decoder, or generative model, samples a latent variable, Z, which is combined with the covariates, C, as its inputs to learn the parameters for a negative binomial generalized linear model (NB-GLM) that maximizes the likelihood of the data.

For a given cell, each gene is modeled using a negative binomial (NB) distribution: $X_{gene} \sim NB(\mu_{gene}, \psi_{gene})$. The NB distribution reflects the discrete nature of mRNA transcripts[37]. Specifically, the NB distribution models the number of successes obtained before r failures for a sequence of independent and identically distributed (IID) Bernoulli trials, where each trial is

analogous to whether a cell expresses a specific mRNA transcript. The NB distribution for each gene is parameterized using a generalized linear model (GLM) for ξ_{gene}, the overdispersion (i.e., the number of failures before stopping), and ψ_{gene}, the log-odds of success.

Specifically, we model the overdispersion per gene with $\xi_{gene} = X_{gene,\mu} * e^{\psi_{gene}}$, where $\ln X_{\mu,gene} = w_{\mu,gene,prop} * x_{gene,prop} + w_{\mu,gene,L} * x_{gene,L} + W_{\mu,gene,cov} * X_{gene,cov} + b_{\mu,gene}$ is the predicted log expression of a gene from the decoder. Here, $X_{gene,prop}$ is the output of the decoder after a softmax transformation, which produces the proportion of transcripts for each gene for a given cell. This proportion is scaled by the library size (total number of transcripts) for each cell, $X_{gene,L}$, which influences the mean gene expression[35]. $X_{gene,cov}$ is a vector of the covariates for a given cell, which include categorical covariates (one-hot encoded) and continuous covariates (z-scored). $W_{\mu,gene,cov}$ is a vector of weights for the covariates, $w_{\mu,gene,prop}$ is a scalar weight for the gene proportion, $w_{\mu,gene,L}$ is a scalar weight for the library size, and the bias term is given by $b_{\mu,gene}$. This differs from scVI's implementation, which does not include a bias term and uses weights for only the primary batch key[9] as opposed to multiple covariates: $\ln X_{\mu,gene} = x_{gene,prop} + x_{gene,L} + W_{\mu,gene,batch} * X_{gene,batch} + b_{\mu,gene}$.

PIANO also models the other NB parameter, the log-odds for capturing a transcript, using $\psi_{gene} = w_{\psi,gene,prop} * x_{gene,prop} + W_{\psi,gene,cov} * X_{gene,cov} + b_{\psi,gene}$. Here, $w_{\psi,gene,prop}$ is the scalar weight for the gene proportion, $W_{\psi,gene,cov}$ is the vector of weights for the covariates, and $b_{\psi,gene}$ is the bias term. We assume that the library size should only impact the mean expression of genes, and therefore, do not include it to model the log-odds. scVI uses a mean-overdispersion parameterization[9], which does not model log-odds.

The likelihood of the data under the PIANO NB-GLM is given by:

$$\mathrm{P}(\mathrm{X}_{\text{cell,gene}} = x_{\text{cell,gene}} \mid \xi_{\text{cell,gene}}, p_{\text{cell,gene}}) = \frac{\Gamma(x_{\text{gene}}+\xi_{\text{gene}})}{\Gamma(x_{\text{gene}}+1)\Gamma(\xi_{\text{gene}})}(1-p_{\text{cell,gene}})^{\xi_{\text{cell,gene}}}(p_{\text{cell,gene}})^{x_{\text{cell,gene}}},$$

where $\mathrm{p}_{\text{cell,gene}} = \frac{e^{\psi_{\text{cell,gene}}}}{1+e^{\psi_{\text{cell,gene}}}}$.

PIANO evidence lower bound objective

PIANO is trained by minimizing an evidence lower bound objective (ELBO), which is a lower bound on the log-probability of the observed gene expression data given the covariates. The ELBO is the sum of a data fidelity term, given by the average log-likelihood of the data under the NB observation model, and the negative Kullback-Leibler divergence (KLD) between the approximate posterior over Z from the recognition model and the isotropic Gaussian prior. The total ELBO is the sum of ELBOs across cells. The ELBO can be derived as follows:

$\log p_\theta(\mathbf{x} \mid \mathbf{c})$	Log-likelihood of the data conditioned on covariates
$= \log \int p_\theta(\mathbf{x}, \mathbf{z} \mid \mathbf{c})\, d\mathbf{z}$	Marginalize over latent variable $\mathbf{z}$
$= \log \int q_\phi(\mathbf{z} \mid \mathbf{x}) \frac{p_\theta(\mathbf{x},\mathbf{z}\mid\mathbf{c})}{q_\phi(\mathbf{z}\mid\mathbf{x})}\, d\mathbf{z}$	Multiply by $1 = q_\phi(\mathbf{z} \mid \mathbf{x})/q_\phi(\mathbf{z} \mid \mathbf{x})$
$= \log \mathbb{E}_{q_\phi(\mathbf{z}\mid\mathbf{x})}\left[\frac{p_\theta(\mathbf{x},\mathbf{z}\mid\mathbf{c})}{q_\phi(\mathbf{z}\mid\mathbf{x})}\right]$	By the definition of expected value
$\geq \mathbb{E}_{q_\phi(\mathbf{z}\mid\mathbf{x})}\left[\log \frac{p_\theta(\mathbf{x},\mathbf{z}\mid\mathbf{c})}{q_\phi(\mathbf{z}\mid\mathbf{x})}\right]$	By Jensen's inequality
$:= \mathcal{L}_{\text{ELBO}}(\mathbf{x}, \mathbf{c})$	Evidence lower Bound objective (ELBO)

The ELBO can be written as an expected log-likelihood and a regularizing KLD term[8].

For a given cell, we have:

$$\mathcal{L}_{\text{ELBO}}(\mathbf{x}, \mathbf{c}) = \mathbb{E}_{q_\phi(\mathbf{z}|\mathbf{x})}\left[\log \frac{p_\theta(\mathbf{x}, \mathbf{z} \mid \mathbf{c})}{q_\phi(\mathbf{z} \mid \mathbf{x})}\right] \quad \text{Evidence lower bound objective (ELBO)}$$

$$= \mathbb{E}_{q_\phi(\mathbf{z}|\mathbf{x})}\left[\log \frac{p_\theta(\mathbf{x} \mid \mathbf{z}, \mathbf{c})\, p(\mathbf{z} \mid \mathbf{c})}{q_\phi(\mathbf{z} \mid \mathbf{x})}\right] \quad \text{By the definition of conditional probability}$$

$$= \mathbb{E}_{q_\phi(\mathbf{z}|\mathbf{x})}\left[\log \frac{p_\theta(\mathbf{x} \mid \mathbf{z}, \mathbf{c})\, p(\mathbf{z})}{q_\phi(\mathbf{z} \mid \mathbf{x})}\right] \quad \text{By the batch correction independence assumption } \mathbf{z} \perp \mathbf{c}$$

$$= \mathbb{E}_{q_\phi(\mathbf{z}|\mathbf{x})}\left[\log p_\theta(\mathbf{x} \mid \mathbf{z}, \mathbf{c}) + \log \frac{p(\mathbf{z})}{q_\phi(\mathbf{z} \mid \mathbf{x})}\right] \quad \text{By the product rule of logarithms}$$

$$= \mathbb{E}_{q_\phi(\mathbf{z}|\mathbf{x})}[\log p_\theta(\mathbf{x} \mid \mathbf{z}, \mathbf{c})] - D_{\text{KL}}(q_\phi(\mathbf{z} \mid \mathbf{x}) \,\|\, p(\mathbf{z})) \quad \text{By the definition of KL divergence}$$

$$= \mathbb{E}_{q_\phi(\mathbf{z}|\mathbf{x})}[\log p_\theta(\mathbf{x} \mid \mathbf{z}, \mathbf{c})] - D_{\text{KL}}(q_\phi(\mathbf{z} \mid \mathbf{x}) \,\|\, \mathcal{N}(\mathbf{0}, \mathbf{I})) \quad \text{Substituting the prior } p(\mathbf{z}) = \mathcal{N}(\mathbf{0}, \mathbf{I})$$

$$= \mathbb{E}_{q_\phi(\mathbf{z}|\mathbf{x})}[\log \text{NB}(\mathbf{x} \mid \boldsymbol{\xi}, \boldsymbol{\psi})] - D_{\text{KL}}(q_\phi(\mathbf{z} \mid \mathbf{x}) \,\|\, \mathcal{N}(\mathbf{0}, \mathbf{I})) \quad \text{Substituting the likelihood } p_\theta(\mathbf{x} \mid \mathbf{z}, \mathbf{c}) = \text{NB}(\mathbf{x} \mid \boldsymbol{\xi}, \boldsymbol{\psi})$$

Here, $\log \text{NB}(\mathbf{x} \mid \boldsymbol{\xi}, \boldsymbol{\psi}) = \sum_{\text{gene}} \left[\log \Gamma(x_{\text{gene}} + \xi_{\text{gene}}) - \log \Gamma(x_{\text{gene}} + 1) - \log \Gamma(\xi_{\text{gene}}) + \xi_{\text{gene}} \log(1 - p_{\text{gene}}) + x_{\text{gene}} \log(p_{\text{gene}})\right]$,
where $\text{p}_{\text{gene}} = \frac{e^{\psi_{\text{gene}}}}{1 + e^{\psi_{\text{gene}}}}$, $\boldsymbol{\xi}$,$\boldsymbol{\psi}$ are the NB-GLM parameters from the generative model using $\mathbf{z}$ and $\mathbf{c}$ as inputs,
and Γ is the Gamma function.

The KLD term is calculated analytically between the encoder's latent space distribution, which is a diagonal Gaussian, and an isotropic Gaussian prior distribution[8]. The log-likelihood is calculated using the log-likelihood of the original data under the NB-GLM with the corresponding covariates. By sampling from the latent space representation, the generative model produces an expectation over the log-likelihood of the data. This VAE model architecture enables PIANO to train in an unsupervised manner.

Implementation details

The encoder, or recognition model, learns an approximate posterior distribution over the latent P(Z|X) (e.g., 32 latent dimensions) given a vector of gene counts (e.g., 4096 highly variable genes); specifically, the encoder outputs a probabilistic posterior latent representation for each cell using an independent mean and standard deviation for each latent dimension. Using the reparameterization trick[8], a z-score is sampled outside of the computational graph to produce the posterior latent representation P(Z|X) from the mean, standard deviation, and z-score. This latent representation is concatenated with covariates as the input to the decoder.

Given a latent vector and a set of covariates (e.g., one-hot encoding for 3 primate species), the decoder parametrizes P(X|Z,C) (i.e., a vector for 4096 genes), which is the proportion of gene counts; specifically, the decoder uses softmax to obtain gene proportions that sum to one. These proportions are then scaled by the library size (total counts) of a given cell to obtain the means of the negative binomial distributions over the genes. Compressing the data through the VAE bottleneck helps the latent space representation capture only the most important biological information for reconstructing the gene proportions. Feeding the covariates into only the decoder helps the model remove batch effects from the latent space while still modeling the original gene counts, which requires information from the covariates.

Both the encoder and decoder are implemented in PyTorch using a multi-layer perceptron (MLP), or feedforward neural network, following the pattern of feed-forward linear layer, batch normalization, rectified linear unit (ReLU), and dropout. PIANO uses a default network architecture with 3 layers, 256 hidden nodes per linear layer, and 32 latent dimensions. PIANO uses default parameters with 1e-5 for the batch normalization epsilon, 1e-1 for

batchnorm momentum, and 0.1 for dropout rate. PIANO uses an AdamW optimizer with a default learning rate of 2e-4, mini-batch size of 128 cells, and no weight decay. For training stability, datasets with normalized total transcript counts in the hundreds of thousands per cell are first mean-scaled to 10,000 transcripts per cell within each batch; the gene expression data are also log1p-transformed before being used as input for the encoder. For each training step, we use a mini-batch update with 128 randomly sampled cells without replacement.

The two most important parameters for PIANO are the number of training epochs and the hyperparameter beta, which govern the tradeoff between biological conservation and batch correction. The number of training epochs increases the training time linearly. Although training costs are not an issue with PIANO, due to its torch.compile GPU acceleration, training the model for too many epochs may result in over-correction for batch effects and risk losing separation of cell types. To mitigate this potential issue, PIANO is designed with easily accessible model checkpoints at fixed training intervals; a visual inspection of the latent space UMAP at each training interval reveals when the model has achieved an acceptable balance between batch correction and biological conservation. Using GPU acceleration from the RAPIDS AI library, UMAP can be computed within seconds even for large datasets. This enables visual inspections to determine whether PIANO has finished training, which helps avoid over-correction of batch effects.

In practice, VAEs are trained using beta-annealing. We scale linearly with every step, where the beta weight is 1.0 after 400 training epochs have completed.

Specifically, we have $\mathcal{L}_{\mathrm{ELBO}}(\mathbf{x}, \mathbf{c}) = \mathbb{E}_{q_\phi(\mathbf{z}|\mathbf{x})}[\log \mathrm{NB}(\mathbf{x} \mid \boldsymbol{\xi}, \boldsymbol{\psi})] - \beta * D_{\mathrm{KL}}(q_\phi(\mathbf{z} \mid \mathbf{x}) \,\|\, \mathcal{N}(\mathbf{0}, \mathbf{I}))$

To mitigate KLD vanishing, PIANO uses a KLD beta-annealing schedule[38] that linearly increases the KLD weight after each mini-batch update. The KLD term is weighted using a beta hyperparameter that increases after each training step; a larger weight generally results in stronger batch correction. PIANO uses a linear KLD annealing schedule starting from an initial weight of 0.0 to its maximum weight after 400 epochs, incrementing after every training step[38]. Although 400 training epochs are usually not necessary, this large upper bound makes the default schedule sufficient for most applications. The max KLD weight is the most important hyperparameter to tune for balancing batch correction with biological conservation of cell types.

We found a max beta of 1.0 to be effective for most datasets. For the lung integration, we used a weaker max beta of 0.5. For most integrations, we used our default model parameters, architecture, and 200 max training epochs (Methods). Additionally, we found our default learning rate of 2e-4 to be effective for the larger datasets, but reduced the learning rate to 1e-4 and used a weight decay of 1e-6 to avoid overfitting the smaller datasets (pancreas, lung, immune, and tabula muris) that had fewer than 50,000 cells; these datasets may have had insufficient training samples for the models to train properly. Moreover, we found our default model architecture design with 3 hidden layers, 256 hidden nodes per hidden layer, and 32 latent dimensions effective on all datasets (Methods). In general, it is sufficient to run PIANO with the default parameters, using model checkpoints to determine if using fewer training epochs are sufficient. If over-correcting for batch effects, PIANO can be run with a lower beta (i.e. 0.5 instead of 1).

Code compilation

PIANO is implemented in PyTorch and achieves faster training through backpropagation compared to existing VAE methods using the torch.compile function. Using a custom training loop, the entire training step is compiled, including both forward and backward passes. This loop also enables model checkpointing without the use of PyTorch Lightning callbacks. A PyTorch DataLoader with a custom Sampler (e.g. GPUBatchSampler) is used to efficiently extract mini-batches from the data, which is stored in a custom PyTorch Dataset class (AnnDataset). This AnnDataset class stores the raw counts and covariates in an augmented data matrix, which supports dense GPU memory mode, a custom sparse GPU memory mode, and CPU memory modes; this allows flexibility depending on the size of the data and availability of GPU memory. These code optimizations enable PIANO to train substantially faster than other VAE methods.

Preprocessing

All integration examples used publicly available and preprocessed datasets; as such, no new quality control thresholds were used. For cross-species comparisons, we subset the data to the human one-to-one orthologs from Ensemble version 114[22]. For the primate basal ganglia integration gene sets, one-to-one ortholog transcription factors were selected from AnimalTFDB4[24] and one-to-one ortholog functional gene sets were queried from UniProtKB[23,39]. Scanpy's sc.pp.highly_variable_genes was used to select 4096 genes for most integrations, using the 'seurat_v3' flavor and the most prominent source of variation for the batch key[31,40]. For the single species dataset integrations in Figure 1, the batch keys used were sequencing platform for pancreas, donor for lung, dataset batch for immune, donor for tabula muris, and donor for heart atlas. For each of the cross-species integrations, species was used as the batch key. For the whole mouse brain spatial integration, all 1,122 genes were used.

For most datasets, the raw gene counts were used as input. For datasets with substantially larger magnitudes for the raw data (i.e. due to normalizing differences in the sequencing data acquisition), each dataset was scaled to a mean of 10k. These datasets include 'Villani' for the immune integration, the 'fluidigmc1', 'smarter', and 'smartseq2' datasets for the pancreas integration, and all donors for the tabula muris integration.

To balance the number of cells between datasets, we used 6400 cells per group for each species in the primate basal ganglia, up to 2^{18} (262,144) cells per species (mouse, human) for each of the Mouse Cell Atlas (MCA) and Human Cell Landscape (HCL), 100 cells per highest resolution cluster for each of the mouse and human whole brain datasets, 6,400 cells per Subclass for each modality (scRNAseq, spatial) in the whole mouse brain integration, and 2^{16} (65,536) cells per developmental timepoint for each species (mouse and marmoset) for the developmental integration.

For principal component analysis (PCA), 30 principal components were used in each integration example. For UMAP visualization, the Scanpy default 15 neighbors were used to compute the neighborhood graph. For Harmony, default parameters were used with max 20 iterations. For Seurat, the k.weight was set to 50 (as opposed to the default 100 to enable integrations on batches with fewer cells). For scVI, the default parameters were used besides the model architecture, which used the same 3 hidden layers, 256 hidden nodes per hidden layer, and 32 latent dimensions as PIANO for benchmarking comparisons.

Benchmarking

Integration embeddings were evaluated by biological conservation metrics, which measure separation of cell type clusters, and batch correction metrics, which measure how mixed cells are between different batches. The biological conservation metrics used were normalized mutual information (NMI) and adjusted Rand index (ARI). The batch correction metrics used were the k-nearest-neighbors batch effect test (KBET) and inverse local Simpson's index (iLISI)[7]. It is well understood that a good integration should maintain separation of cell type clusters while mixing data between batches without over correction[7].

The scib_metrics package was used to run integration benchmarking with commonly used metrics[7] The average of the Leiden Adjusted Rand Index (ARI) and Leiden Normalized Mutual Information (NMI) scores was used to measure biological conservation; both metrics use de novo Leiden clustering on the latent space representation of the data and compare the clusters to the ground truth labels[7]. The average of the K-nearest neighbors Batch Effect Test (KBET) and integration Local Inverse Simpson's Index (iLISI) scores were used to quantify batch correction; these metrics compare the number of cells from different batches in the neighbors of a given cell[7].

Code availability

The PIANO source code is publicly available at https://github.com/NingWang1729/piano with an installable package at https://pypi.org/project/piano-integration. A publicly available Code Ocean capsule demonstrating the human and mouse whole brain integration is available at https://codeocean.allenneuraldynamics.org/capsule/4212260/tree/v2.

Data availability

All datasets used were already preprocessed and publicly available. Download links can be found in the key resources table.

Author Contributions

N.W., C.L., J.W.P., and F.M.K. designed the method. N.W. and D.T. implemented the method. N.W., H.F., and V.N. benchmarked other methods. N.W., H.F., V.N., N.S., C.L., D.Y., Z.Y., C.C., M.D., S.D., L.C., and J.S. performed data collection, preprocessing, and analysis. N.W., C.L., J.W.P., and F.M.K. wrote the manuscript with input from all authors.

REFERENCES

1. Siletti, K., Hodge, R., Mossi, A.A., Lee, K.W., Ding, S.L., Hu, L., Lönnerberg, P., Bakken, T., Casper, T., Clark, M., et al. (2023). Transcriptomic diversity of cell types across the adult human brain. Science (New York, N.Y.) *382*. https://doi.org/10.1126/science.add7046.

2. Yao, Z., van Velthoven, C.T.J., Kunst, M., Zhang, M., McMillen, D., Lee, C., Jung, W., Goldy, J., Abdelhak, A., Aitken, M., et al. (2023). A high-resolution transcriptomic and spatial atlas of cell types in the whole mouse brain. Nature *624*, 317–332.

3. Langlieb, J., Sachdev, N.S., Balderrama, K.S., Nadaf, N.M., Raj, M., Murray, E., Webber, J.T., Vanderburg, C., Gazestani, V., Tward, D., et al. (2023). The molecular cytoarchitecture of the adult mouse brain. Nature *624*, 333–342.

4. Krienen, F.M., Levandowski, K.M., Zaniewski, H., del Rosario, R.C.H., Schroeder, M.E., Goldman, M., Wienisch, M., Lutservitz, A., Beja-Glasser, V.F., Chen, C., et al. (2023). A marmoset brain cell census reveals regional specialization of cellular identities. Science Advances. https://doi.org/10.1126/sciadv.adk3986.

5. Korsunsky, I., Millard, N., Fan, J., Slowikowski, K., Zhang, F., Wei, K., Baglaenko, Y., Brenner, M., Loh, P.-R., and Raychaudhuri, S. (2019). Fast, sensitive and accurate integration of single-cell data with Harmony. Nature Methods *16*, 1289–1296.

6. Butler, A., Hoffman, P., Smibert, P., Papalexi, E., and Satija, R. (2018). Integrating single-cell transcriptomic data across different conditions, technologies, and species. Nature Biotechnology *36*, 411–420.

7. Luecken, M.D., Büttner, M., Chaichoompu, K., Danese, A., Interlandi, M., Mueller, M.F., Strobl, D.C., Zappia, L., Dugas, M., Colomé-Tatché, M., et al. (2022). Benchmarking atlas-level data integration in single-cell genomics. Nat Methods *19*, 41–50.

8. Kingma, D.P., and Welling, M. (2013). Auto-Encoding Variational Bayes.

9. Lopez, R., Regier, J., Cole, M.B., Jordan, M.I., and Yosef, N. (2018). Deep generative modeling for single-cell transcriptomics. Nature Methods *15*, 1053–1058.

10. Vieira Braga, F.A., Kar, G., Berg, M., Carpaij, O.A., Polanski, K., Simon, L.M., Brouwer, S., Gomes, T., Hesse, L., Jiang, J., et al. (2019). A cellular census of human lungs identifies novel cell states in health and in asthma. Nature Medicine *25*, 1153–1163.

11. Grün, D., Muraro, M.J., Boisset, J.-C., Wiebrands, K., Lyubimova, A., Dharmadhikari, G., van den Born, M., van Es, J., Jansen, E., Clevers, H., et al. (2016). De Novo Prediction of Stem Cell Identity using Single-Cell Transcriptome Data. Cell Stem Cell *19*, 266–277.

12. Muraro, M.J., Dharmadhikari, G., Grün, D., Groen, N., Dielen, T., Jansen, E., van Gurp, L., Engelse, M.A., Carlotti, F., de Koning, E.J.P., et al. (2016). A Single-Cell Transcriptome Atlas of the Human Pancreas. Cell Syst *3*, 385–394.e3.

13. Lawlor, N., George, J., Bolisetty, M., Kursawe, R., Sun, L., Sivakamasundari, V., Kycia, I., Robson, P., and Stitzel, M.L. (2017). Single-cell transcriptomes identify human islet cell signatures and reveal cell-type-specific expression changes in type 2 diabetes. Genome Res *27*, 208–222.

14. Segerstolpe, Å., Palasantza, A., Eliasson, P., Andersson, E.-M., Andréasson, A.-C., Sun, X., Picelli, S., Sabirsh, A., Clausen, M., Bjursell, M.K., et al. (2016). Single-Cell Transcriptome Profiling of Human Pancreatic Islets in Health and Type 2 Diabetes. Cell Metab *24*, 593–607.

15. Kanemaru, K., Cranley, J., Muraro, D., Miranda, A.M.A., Ho, S.Y., Wilbrey-Clark, A., Patrick Pett, J., Polanski, K., Richardson, L., Litvinukova, M., et al. (2023). Spatially resolved multiomics of human cardiac niches. Nature *619*, 801–810.

16. Oetjen, K.A., Lindblad, K.E., Goswami, M., Gui, G., Dagur, P.K., Lai, C., Dillon, L.W., McCoy, J.P., and Hourigan, C.S. (2018). Human bone marrow assessment by single-cell RNA sequencing, mass cytometry, and flow cytometry. JCI Insight *3*. https://doi.org/10.1172/jci.insight.124928.

17. Datasets -Single Cell Gene Expression -Official 10x Genomics Support https://support.10xgenomics.com/single-cell-gene-expression/datasets/3.0.0/pbmc_10k_v3.

18. Freytag, S., Tian, L., Lönnstedt, I., Ng, M., and Bahlo, M. (2018). Comparison of clustering tools in R for medium-sized 10x Genomics single-cell RNA-sequencing data. F1000Research *7*. https://doi.org/10.12688/f1000research.15809.2.

19. Sun, Z., Chen, L., Xin, H., Jiang, Y., Huang, Q., Cillo, A.R., Tabib, T., Kolls, J.K., Bruno, T.C., Lafyatis, R., et al. (2019). A Bayesian mixture model for clustering droplet-based single-cell transcriptomic data from population studies. Nature Communications *10*, 1–10.

20. Villani, A.-C., Satija, R., Reynolds, G., Sarkizova, S., Shekhar, K., Fletcher, J., Griesbeck, M., Butler, A., Zheng, S., Lazo, S., et al. (2017). Single-cell RNA-seq reveals new types of human blood dendritic cells, monocytes, and progenitors. Science *356*. https://doi.org/10.1126/science.aah4573.

21. Single-cell transcriptomics of 20 mouse organs creates a Tabula Muris (2018). Nature *562*, 367–372.

22. Dyer, S.C., Austine-Orimoloye, O., Azov, A.G., Barba, M., Barnes, I., Barrera-Enriquez, V.P., Becker, A., Bennett, R., Beracochea, M., Berry, A., et al. (2025). Ensembl 2025. Nucleic Acids Res *53*, D948–D957.

23. Ahmad, S., Bowler-Barnett, E.H., Rice, D.L., Kim, M., Wijerathne, S., Luciani, A., Kandasaamy, S., Luo, J., Watkins, X., Turner, E., et al. (2025). The UniProt website API: facilitating programmatic access to protein knowledge. Nucleic Acids Res *53*, W547–W553.

24. Shen, W.K., Chen, S.Y., Gan, Z.Q., Zhang, Y.Z., Yue, T., Chen, M.M., Xue, Y., Hu, H., and Guo, A.Y. (2023). AnimalTFDB 4.0: a comprehensive animal transcription factor database updated with variation and expression annotations. Nucleic acids research *51*. https://doi.org/10.1093/nar/gkac907.

25. Bendesky, A. (2025). Behavioral evolution by diverging cell type composition. Current opinion in genetics & development *95*. https://doi.org/10.1016/j.gde.2025.102397.

26. Schroeder, M.E., McCormack, D.M., Metzner, L.R., Kang, J., Li, K.X., Yu, E., Melamed, L., Levandowski, K.M., Zaniewski, H., Zhang, Q., et al. (2025). Astrocyte regional specialization is shaped by postnatal development. bioRxiv, 2024.10.11.617802. https://doi.org/10.1101/2024.10.11.617802.

27. Han, X., Wang, R., Zhou, Y., Fei, L., Sun, H., Lai, S., Saadatpour, A., Zhou, Z., Chen, H., Ye, F., et al. (2018). Mapping the Mouse Cell Atlas by Microwell-Seq. Cell *173*, 1307.

28. Han, X., Zhou, Z., Fei, L., Sun, H., Wang, R., Chen, Y., Chen, H., Wang, J., Tang, H., Ge, W., et al. (2020). Construction of a human cell landscape at single-cell level. Nature *581*, 303–309.

29. Chen, K.H., Boettiger, A.N., Moffitt, J.R., Wang, S., and Zhuang, X. (2015). Spatially resolved, highly multiplexed RNA profiling in single cells. Science. https://doi.org/10.1126/science.aaa6090.

30. Zhang, M., Pan, X., Jung, W., Halpern, A.R., Eichhorn, S.W., Lei, Z., Cohen, L., Smith, K.A., Tasic, B., Yao, Z., et al. (2023). Molecularly defined and spatially resolved cell atlas of the whole mouse brain. Nature *624*, 343–354.

31. Stuart, T., Butler, A., Hoffman, P., Hafemeister, C., Papalexi, E., Mauck, W.M., 3rd, Hao, Y., Stoeckius, M., Smibert, P., and Satija, R. (2019). Comprehensive Integration of Single-Cell Data. Cell *177*, 1888–1902.e21.

32. Satija, R., Farrell, J.A., Gennert, D., Schier, A.F., and Regev, A. (2015). Spatial reconstruction of single-cell gene expression data. Nature Biotechnology *33*, 495–502.

33. Murray, R. (2011). Skillful writing of an awful research paper. https://doi.org/10.1021/ac2000169.

34. Hrovatin, K., Moinfar, A.A., Zappia, L., Lapuerta, A.T., Lengerich, B., Kellis, M., and Theis, F.J. (2024). Integrating single-cell RNA-seq datasets with substantial batch effects. bioRxiv : the preprint server for biology. https://doi.org/10.1101/2023.11.03.565463.

35. Choudhary, S., and Satija, R. (2022). Comparison and evaluation of statistical error models for scRNA-seq. Genome Biology *23*, 1–20.

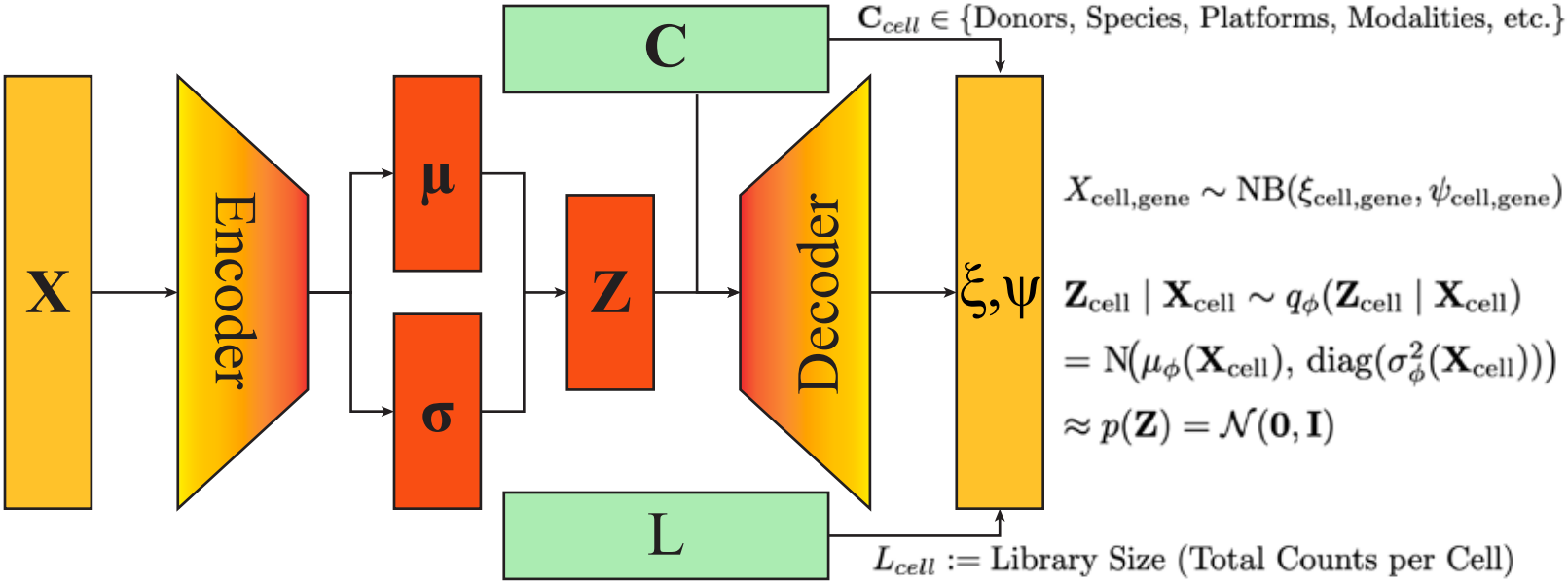
36. Xu, C., Lopez, R., Mehlman, E., Regier, J., Jordan, M.I., and Yosef, N. (2021). Probabilistic harmonization and annotation of single‐cell transcriptomics data with deep generative models. Molecular Systems Biology. https://doi.org/10.15252/msb.20209620.

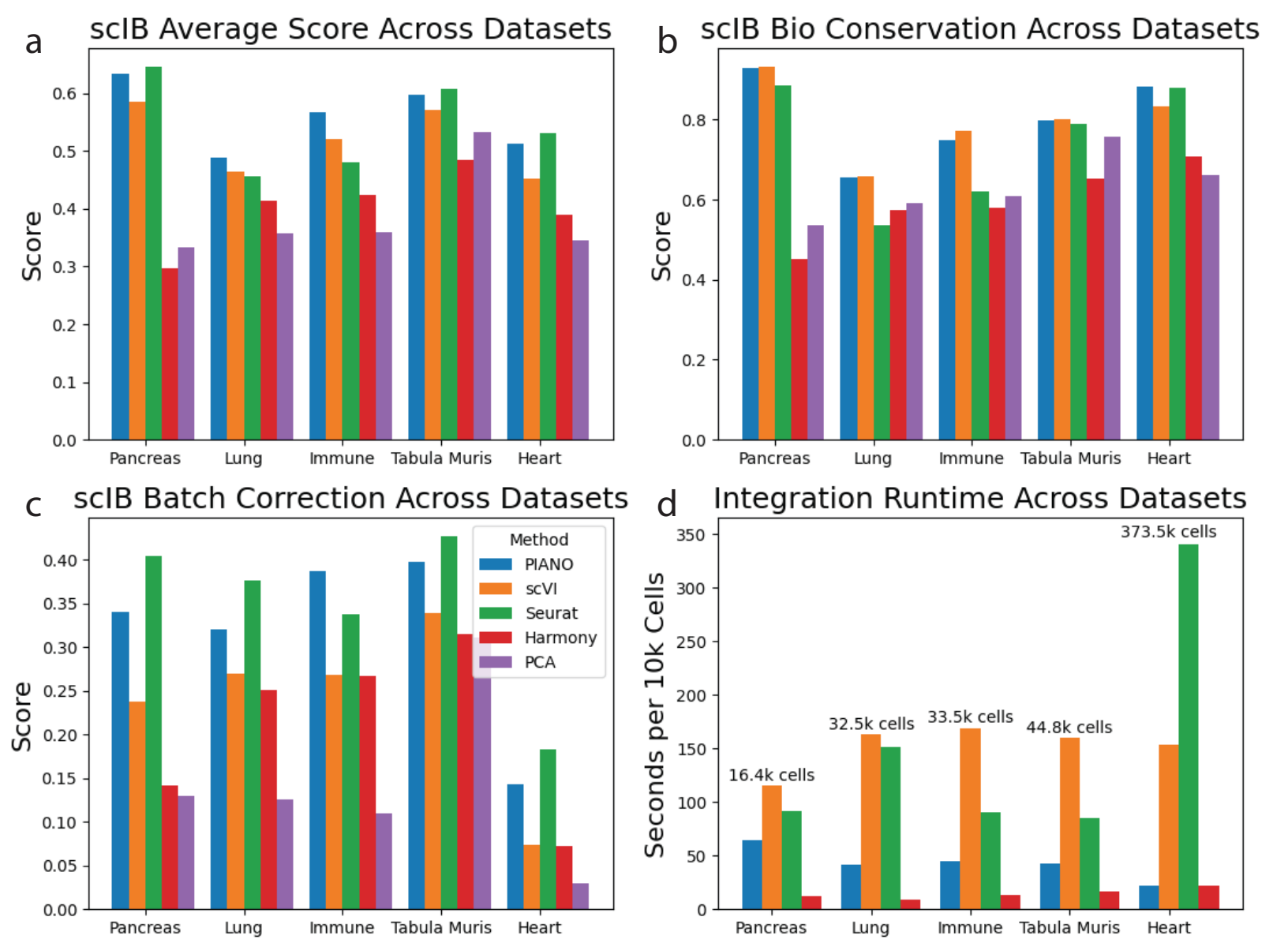
37. Svensson, V. (2020). Droplet scRNA-seq is not zero-inflated. Nature Biotechnology *38*, 147–150.

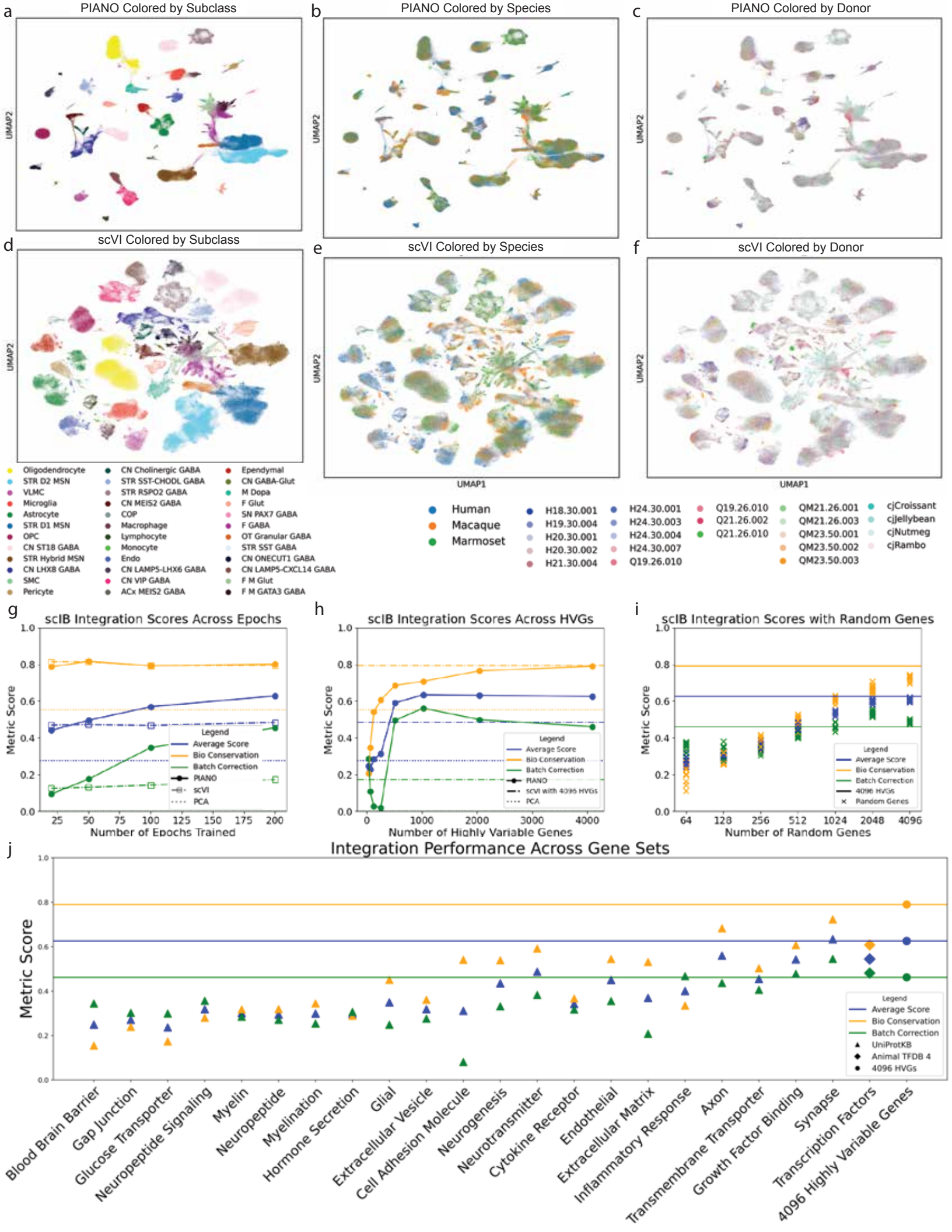
38. Fu, H., Li, C., Liu, X., Gao, J., Celikyilmaz, A., and Carin, L. (2019). Cyclical Annealing Schedule: A Simple Approach to Mitigating KL Vanishing.

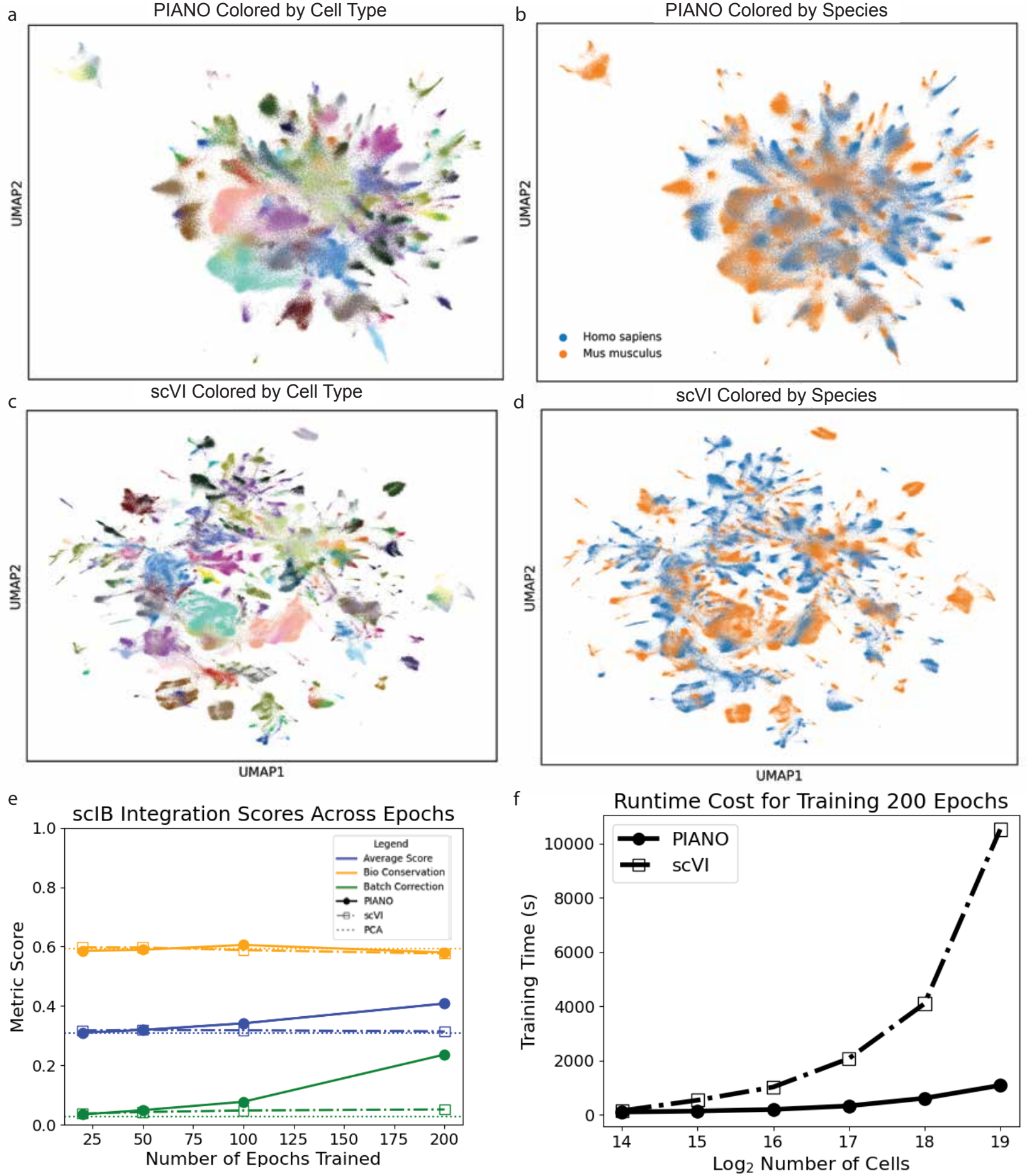
39. The UniProt Consortium (2025). UniProt: the Universal Protein Knowledgebase in 2025. Nucleic Acids Res. *53*, D609–D617.

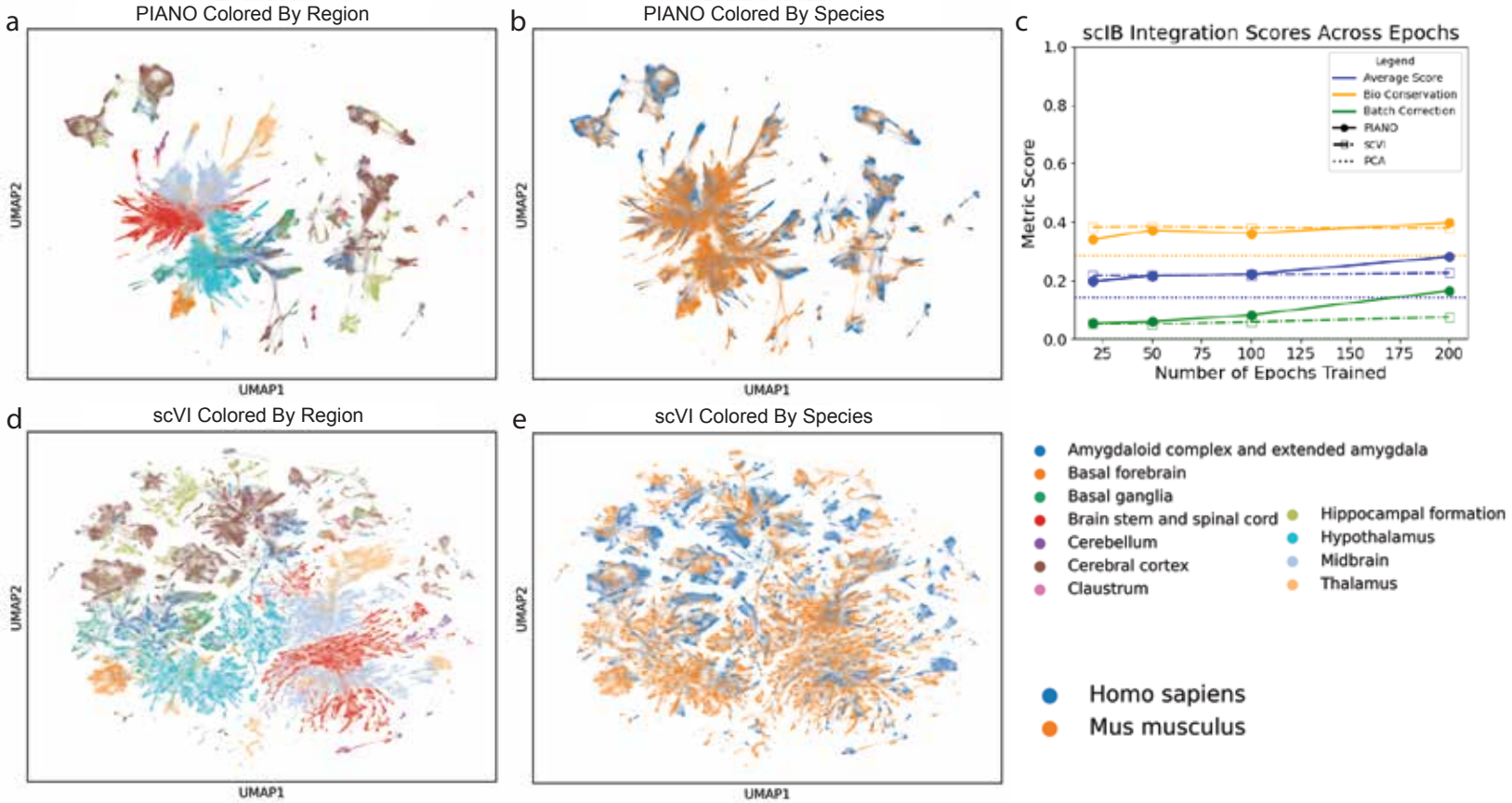
40. Wolf, F.A., Angerer, P., and Theis, F.J. (2018). SCANPY: large-scale single-cell gene expression data analysis. Genome Biology *19*, 1–5.

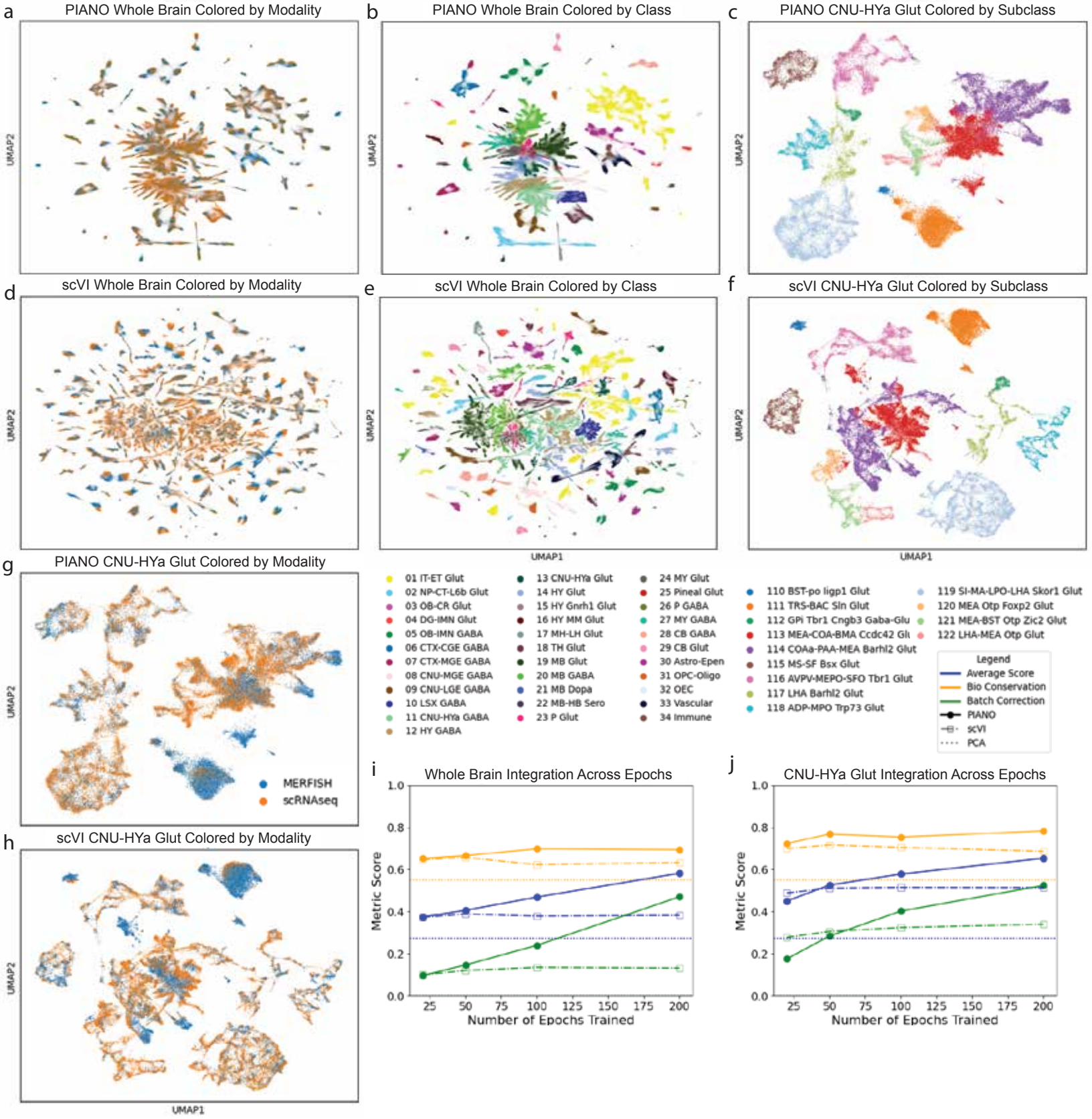

$\mathbf{C}_{cell} \in$ {Donors, Species, Platforms, Modalities, etc.}
C
X
Encoder
μ
σ
Z
Decoder
ξ,ψ
$X_{\text{cell,gene}} \sim \text{NB}(\xi_{\text{cell,gene}}, \psi_{\text{cell,gene}})$
$\mathbf{Z}_{\text{cell}} \mid \mathbf{X}_{\text{cell}} \sim q_\phi(\mathbf{Z}_{\text{cell}} \mid \mathbf{X}_{\text{cell}})$
$= \text{N}\big(\mu_\phi(\mathbf{X}_{\text{cell}}), \text{diag}(\sigma^2_\phi(\mathbf{X}_{\text{cell}}))\big)$
$\approx p(\mathbf{Z}) = \mathcal{N}(\mathbf{0}, \mathbf{I})$
L
L_{cell} := Library Size (Total Counts per Cell)

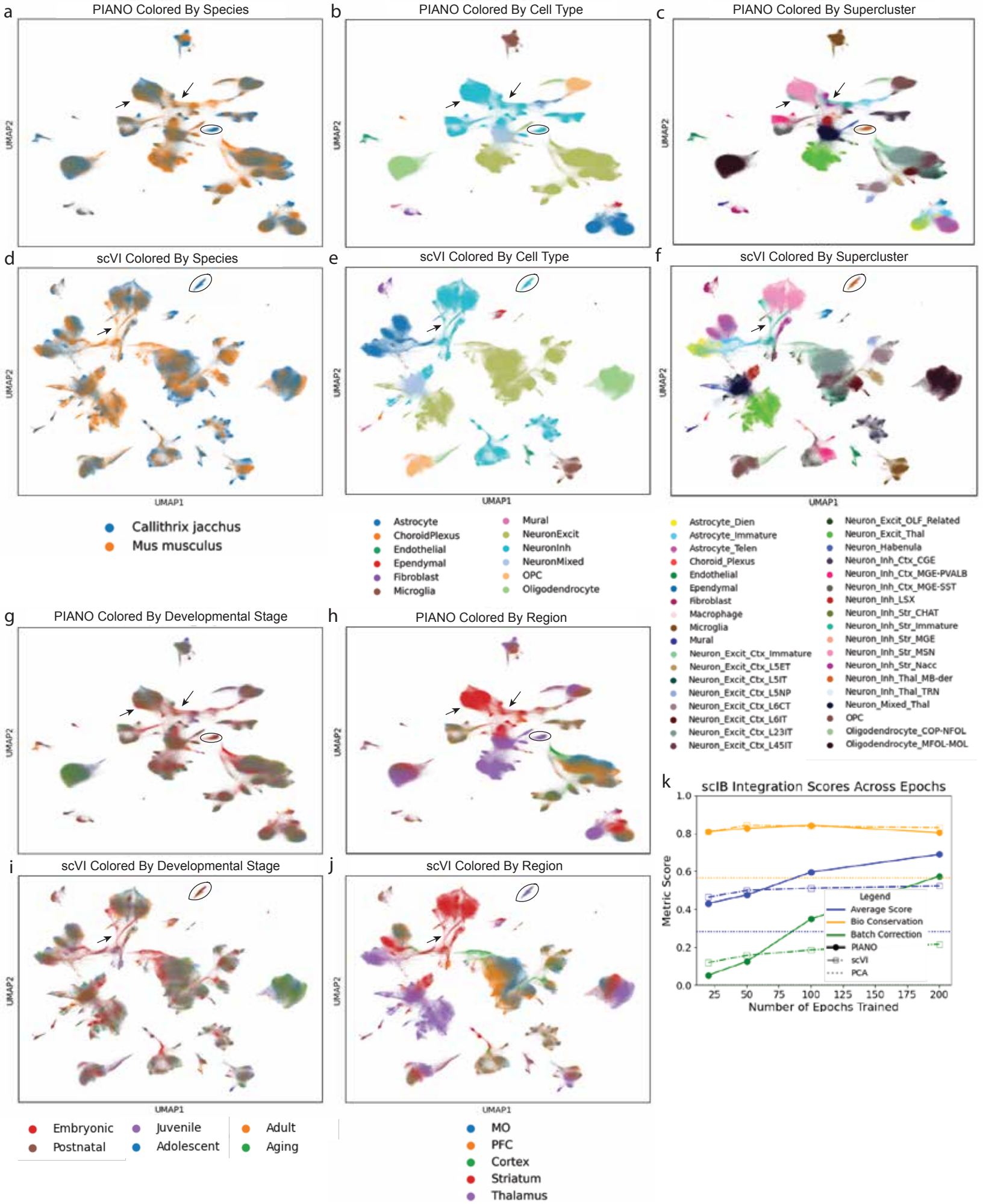

a
scIB Average Score Across Datasets
Score
0.0
0.1
0.2
0.3
0.4
0.5
0.6
Pancreas
Lung
Immune
Tabula Muris
Heart
b
scIB Bio Conservation Across Datasets
Score
0.0
0.2
0.4
0.6
0.8
Pancreas
Lung
Immune
Tabula Muris
Heart
c
scIB Batch Correction Across Datasets
Score
0.00
0.05
0.10
0.15
0.20
0.25
0.30
0.35
0.40
Method
PIANO
scVI
Seurat
Harmony
PCA
Pancreas
Lung
Immune
Tabula Muris
Heart
d
Integration Runtime Across Datasets
Seconds per 10k Cells
0
50
100
150
200
250
300
350
16.4k cells
32.5k cells
33.5k cells
44.8k cells
373.5k cells
Pancreas
Lung
Immune
Tabula Muris
Heart


a
PIANO Colored by Subclass
b
PIANO Colored by Species
c
PIANO Colored by Donor
d
scVI Colored by Subclass
e
scVI Colored by Species
f
scVI Colored by Donor
UMAP1
UMAP2
Oligodendrocyte
STR D2 MSN
VLMC
Microglia
Astrocyte
STR D1 MSN
OPC
CN ST18 GABA
STR Hybrid MSN
CN LHX8 GABA
SMC
Pericyte
CN Cholinergic GABA
STR SST-CHODL GABA
STR RSPO2 GABA
CN MEIS2 GABA
COP
Macrophage
Lymphocyte
Monocyte
Endo
CN LAMP5-LHX6 GABA
CN VIP GABA
ACx MEIS2 GABA
Ependymal
CN GABA-Glut
M Dopa
F Glut
SN PAX7 GABA
F GABA
OT Granular GABA
STR SST GABA
CN ONECUT1 GABA
CN LAMP5-CXCL14 GABA
F M Glut
F M GATA3 GABA
Human
Macaque
Marmoset
H18.30.001
H19.30.004
H20.30.001
H20.30.002
H21.30.004
H24.30.001
H24.30.003
H24.30.004
H24.30.007
Q19.26.010
Q19.26.010
Q21.26.002
Q21.26.010
QM21.26.001
QM21.26.003
QM23.50.001
QM23.50.002
QM23.50.003
cjCroissant
cjJellybean
cjNutmeg
cjRambo
g
scIB Integration Scores Across Epochs
Metric Score
Number of Epochs Trained
Legend
Average Score
Bio Conservation
Batch Correction
PIANO
scVI
PCA
h
scIB Integration Scores Across HVGs
Metric Score
Number of Highly Variable Genes
Legend
Average Score
Bio Conservation
Batch Correction
PIANO
scVI with 4096 HVGs
PCA
i
scIB Integration Scores with Random Genes
Metric Score
Number of Random Genes
Legend
Average Score
Bio Conservation
Batch Correction
4096 HVGs
Random Genes
j
Integration Performance Across Gene Sets
Metric Score
Legend
Average Score
Bio Conservation
Batch Correction
UniProtKB
Animal TFDB 4
4096 HVGs
Blood Brain Barrier
Gap Junction
Glucose Transporter
Neuropeptide Signaling
Myelin
Neuropeptide
Myelination
Hormone Secretion
Glial
Extracellular Vesicle
Cell Adhesion Molecule
Neurogenesis
Neurotransmitter
Cytokine Receptor
Endothelial
Extracellular Matrix
Inflammatory Response
Axon
Transmembrane Transporter
Growth Factor Binding
Synapse
Transcription Factors
4096 Highly Variable Genes


a
PIANO Colored by Cell Type
UMAP2
b
PIANO Colored by Species
UMAP2
Homo sapiens
Mus musculus
c
scVI Colored by Cell Type
UMAP2
UMAP1
d
scVI Colored by Species
UMAP2
UMAP1
e
scIB Integration Scores Across Epochs
Legend
Average Score
Bio Conservation
Batch Correction
PIANO
scVI
PCA
Metric Score
1.0
0.8
0.6
0.4
0.2
0.0
25
50
75
100
125
150
175
200
Number of Epochs Trained
f
Runtime Cost for Training 200 Epochs
PIANO
scVI
Training Time (s)
10000
8000
6000
4000
2000
0
14
15
16
17
18
19
Log2 Number of Cells


a
PIANO Colored By Region
UMAP2
UMAP1
b
PIANO Colored By Species
UMAP2
UMAP1
c
scIB Integration Scores Across Epochs
Legend
Average Score
Bio Conservation
Batch Correction
PIANO
SCVI
PCA
Metric Score
1.0
0.8
0.6
0.4
0.2
0.0
25
50
75
100
125
150
175
200
Number of Epochs Trained
d
scVI Colored By Region
UMAP2
UMAP1
e
scVI Colored By Species
UMAP2
UMAP1
Amygdaloid complex and extended amygdala
Basal forebrain
Basal ganglia
Brain stem and spinal cord
Cerebellum
Cerebral cortex
Claustrum
Hippocampal formation
Hypothalamus
Midbrain
Thalamus
Homo sapiens
Mus musculus


a
PIANO Whole Brain Colored by Modality
b
PIANO Whole Brain Colored by Class
c
PIANO CNU-HYa Glut Colored by Subclass
d
scVI Whole Brain Colored by Modality
e
scVI Whole Brain Colored by Class
f
scVI CNU-HYa Glut Colored by Subclass
g
PIANO CNU-HYa Glut Colored by Modality
MERFISH
scRNAseq
h
scVI CNU-HYa Glut Colored by Modality
UMAP1
UMAP2
01 IT-ET Glut
02 NP-CT-L6b Glut
03 OB-CR Glut
04 DG-IMN Glut
05 OB-IMN GABA
06 CTX-CGE GABA
07 CTX-MGE GABA
08 CNU-MGE GABA
09 CNU-LGE GABA
10 LSX GABA
11 CNU-HYa GABA
12 HY GABA
13 CNU-HYa Glut
14 HY Glut
15 HY Gnrh1 Glut
16 HY MM Glut
17 MH-LH Glut
18 TH Glut
19 MB Glut
20 MB GABA
21 MB Dopa
22 MB-HB Sero
23 P Glut
24 MY Glut
25 Pineal Glut
26 P GABA
27 MY GABA
28 CB GABA
29 CB Glut
30 Astro-Epen
31 OPC-Oligo
32 OEC
33 Vascular
34 Immune
110 BST-po Iigp1 Glut
111 TRS-BAC Sln Glut
112 GPi Tbr1 Cngb3 Gaba-Glu
113 MEA-COA-BMA Ccdc42 Gl
114 COAa-PAA-MEA Barhl2 Glut
115 MS-SF Bsx Glut
116 AVPV-MEPO-SFO Tbr1 Glut
117 LHA Barhl2 Glut
118 ADP-MPO Trp73 Glut
119 SI-MA-LPO-LHA Skor1 Glut
120 MEA Otp Foxp2 Glut
121 MEA-BST Otp Zic2 Glut
122 LHA-MEA Otp Glut
Legend
Average Score
Bio Conservation
Batch Correction
PIANO
scVI
PCA
i
Whole Brain Integration Across Epochs
j
CNU-HYa Glut Integration Across Epochs
Metric Score
Number of Epochs Trained
0.0
0.2
0.4
0.6
0.8
1.0
25
50
75
100
125
150
175
200


a
PIANO Colored By Species
b
PIANO Colored By Cell Type
c
PIANO Colored By Supercluster
d
scVI Colored By Species
e
scVI Colored By Cell Type
f
scVI Colored By Supercluster
UMAP1
UMAP2
Callithrix jacchus
Mus musculus
Astrocyte
ChoroidPlexus
Endothelial
Ependymal
Fibroblast
Microglia
Mural
NeuronExcit
NeuronInh
NeuronMixed
OPC
Oligodendrocyte
Astrocyte_Dien
Astrocyte_Immature
Astrocyte_Telen
Choroid_Plexus
Endothelial
Ependymal
Fibroblast
Macrophage
Microglia
Mural
Neuron_Excit_Ctx_Immature
Neuron_Excit_Ctx_L5ET
Neuron_Excit_Ctx_L5IT
Neuron_Excit_Ctx_L5NP
Neuron_Excit_Ctx_L6CT
Neuron_Excit_Ctx_L6IT
Neuron_Excit_Ctx_L23IT
Neuron_Excit_Ctx_L45IT
Neuron_Excit_OLF_Related
Neuron_Excit_Thal
Neuron_Habenula
Neuron_Inh_Ctx_CGE
Neuron_Inh_Ctx_MGE-PVALB
Neuron_Inh_Ctx_MGE-SST
Neuron_Inh_LSX
Neuron_Inh_Str_CHAT
Neuron_Inh_Str_Immature
Neuron_Inh_Str_MGE
Neuron_Inh_Str_MSN
Neuron_Inh_Str_Nacc
Neuron_Inh_Thal_MB-der
Neuron_Inh_Thal_TRN
Neuron_Mixed_Thal
OPC
Oligodendrocyte_COP-NFOL
Oligodendrocyte_MFOL-MOL
g
PIANO Colored By Developmental Stage
h
PIANO Colored By Region
i
scVI Colored By Developmental Stage
j
scVI Colored By Region
Embryonic
Postnatal
Juvenile
Adolescent
Adult
Aging
MO
PFC
Cortex
Striatum
Thalamus
k
scIB Integration Scores Across Epochs
Metric Score
Number of Epochs Trained
Legend
Average Score
Bio Conservation
Batch Correction
PIANO
scVI
PCA