This is the author's accepted manuscript of L**uecken, M.D., Gigante, S., Burkhardt, D.B. et al. Defining and benchmarking open problems in single-cell analysis. Nat Biotechnol 43, 1035–1040 (2025),** published in Nature Biotechnology. This version of the article has been accepted for publication, after peer review (when applicable) and is subject to Springer Nature's AM terms of use, but is not the Version of Record and does not reflect post-acceptance improvements, or any corrections. The Version of Record is available online at: https://doi.org/10.1038/s41587-025-02694-w

Defining and benchmarking open problems in single-cell analysis

Malte D. Luecken[1,2*], Scott Gigante[3*], Daniel B. Burkhardt[4*], Robrecht Cannoodt[5,6,7*], Daniel C. Strobl[1,8,9^], Nikolay S. Markov[10^], Luke Zappia[1,5,11^], Giovanni Palla[1,9^], Wesley Lewis[12^], Daniel Dimitrov[13^], Michael E. Vinyard[14,15,16^], D.S. Magruder[17^], Michaela F. Mueller[1,2,9^], Alma Andersson[18,19,20], Emma Dann[21], Qian Qin[15], Dominik J. Otto[22,23,24], Michal Klein[25], Olga Borisovna Botvinnik[26,27], Louise Deconinck[6,7], Kai Waldrant[5], Sai Nirmayi Yasa[5], Artur Szałata[1,11], Andrew Benz[4], Zhijian Li[15,16], The Open Problems Jamboree Members, Jonathan M. Bloom[28], Angela Oliveira Pisco[26,29], Julio Saez-Rodriguez[13], Drausin Wulsin[3], Luca Pinello[16], Yvan Saeys[6,7,30], Fabian J Theis[1,11,31 †], Smita Krishnaswamy[12,17,32 †]

[1] Institute of computational Biology, Helmholtz Munich, Neuherberg, Germany
[2] Institute of Lung Health & Immunity, Helmholtz Munich; Member of the German Center for Lung Research (DZL), Munich, Germany.
[3] Immunai, New York, USA
[4] Cellarity, Inc. Somerville, USA
[5] Data Intuitive, Lebbeke, Belgium
[6] Data Mining and Modelling for Biomedicine group, VIB Center for Inflammation Research, Ghent, Belgium
[7] Department of Applied Mathematics, Computer Science, and Statistics, Ghent University, Ghent, Belgium
[8] Institute of Clinical Chemistry and Pathobiochemistry, School of Medicine, Technical University of Munich, Munich, Germany
[9] TUM School of Life Sciences Weihenstephan, Technical University of Munich, Germany
[10] Division of Pulmonary and Critical Care Medicine, Feinberg School of Medicine, Northwestern University
[11] Department of Mathematics, School of Computing, Information and Technology, Technical University of Munich, Munich, Germany
[12] Interdepartmental Program in Computational Biology and Bioinformatics, Yale University, New Haven, CT 06511, USA
[13] Heidelberg University, Faculty of Medicine, and Heidelberg University Hospital, Institute for Computational Biomedicine, Heidelberg, Germany
[14] Department of Chemistry and Chemical Biology, Harvard University, Cambridge, MA, USA
[15] Broad Institute of Harvard and MIT, Cambridge, MA, USA
[16] Molecular Pathology Unit, Center for Cancer Research, Massachusetts General Hospital, Boston, MA, USA
[17] Department of Computer Science, Yale University, New Haven CT, USA
[18] Genentech Inc
[19] Royal Institute of Technology (KTH), Gene Technology
[20] Science for Life Laboratory (SciLifeLab)
[21] Wellcome Sanger Institute, Wellcome Genome Campus, Cambridge, UK
[22] Basic Sciences Division, Fred Hutchinson Cancer Center, Seattle WA
[23] Computational Biology Program, Public Health Sciences Division, Seattle WA
[24] Translational Data Science IRC, Fred Hutchinson Cancer Center, Seattle WA

[25] Apple
[26] Data Sciences Platform, Chan Zuckerberg Biohub, 499 Illinois St, San Francisco, CA 94158
[27] Bridge Bio Pharma, 3160 Porter Drive, Suite 250, Palo Alto, CA, 94304
[28] Massachusetts Institute of Technology
[29] Insitro, South San Francisco
[30] VIB Center for AI & Computational Biology (VIB.AI), Gent, Belgium
[31] Cellular Genetics Programme, Wellcome Sanger Institute, Hinxton, UK (associated faculty)
[32] Department of Genetics, Yale University, New Haven CT, USA

*,^, † Equal contribution
Correspondence to: smita.krishnaswamy@yale.edu, fabian.theis@helmholtz-munich.de

With the growing number of single-cell analysis tools, benchmarks are increasingly important to guide analysis and method development. However, a lack of standardisation and extensibility in current benchmarks limits their usability, longevity, and relevance to the community. We present Open Problems, a living, extensible, community-guided benchmarking platform including 12 current single-cell tasks that we envision will raise standards for the selection, evaluation, and development of methods in single-cell analysis.

Single-cell genomics has enabled the study of biological processes at an unprecedented scale and resolution[1–3]. These studies were enabled by innovative data generation technologies coupled with emerging computational tools specialised for single-cell data. As single-cell technologies have become more prevalent, so has the development of new analysis tools, which have resulted in over 1700 published algorithms[4] (as of February 2024). Thus, there is an increasing need to continuously evaluate which algorithm performs best in which context to inform best practices[5,6] that evolve with the field.

In many fields of quantitative science, public competitions and benchmarks address this need by evaluating state-of-the-art methods against known criteria, following the concept of a Common Task Framework[7]. Public competitions of this kind have a rich track record of accelerating innovation in algorithm development in computer vision (ImageNet[8]), natural language processing (GLUE[9]), robotics (RoboCup[10]), recommendation systems (Netflix Challenge[11]), and, more recently in the life sciences, in protein structure prediction (CASP[12]) or systems biology (DREAM[13]).

In single-cell genomics, as in many other domains, it is typical for analysis algorithms to be evaluated via benchmarks. However, such benchmarks are often of limited use as the field suffers from a lack of standardised procedures for benchmarking[14], which can lead to different assessments of the same method producing different outcomes. Bespoke benchmarks set up by method developers to evaluate newly developed algorithms often include datasets and metrics chosen to highlight the advantage of their tools, which has been shown to lead to less objective assessments[15,16]. Even if datasets and metrics are standardised, historical analysis

shows that when benchmarks are implemented by the same groups introducing new methods, the evaluations tend to inflate performance of the newest models via custom hyperparameter selection and data processing[17]. To provide more uniform and neutral assessment, groups can perform specialised benchmarking studies independently of method development. Tools such as registered reports, that promote neutrality of benchmarking results by design[18], have recently gained in popularity to enable such studies. These efforts aim to systematically evaluate the current state of the art in a given area[19–22] and may be less biased. However, their results are static and inevitably age. These frameworks are typically not designed for extensibility or interoperability, limiting the value of reusing a framework to perform new systematic benchmarks[14]. This inability to reuse infrastructure leads to repeats of non-standardized benchmarks that cannot provide the guidance that users need. For example, at least four benchmarks of batch integration methods exist[20,23–25], each of which uses different sets of datasets and metrics and thus suggest different optimal methods (**Fig 1a**). Similar issues have been reported across other single-cell topics, where datasets and metrics typically have less than 10% overlap between benchmarks[26]. Ideally, benchmarks that guide users and promote method innovation use consistently applied datasets and metrics that are established independently of method development and with ongoing community participation[14,15,26]. Such community participation around quantified tasks requires continuous updates, a process that is hard to realise in the typical result-paper framework that defines the modern scientific process.

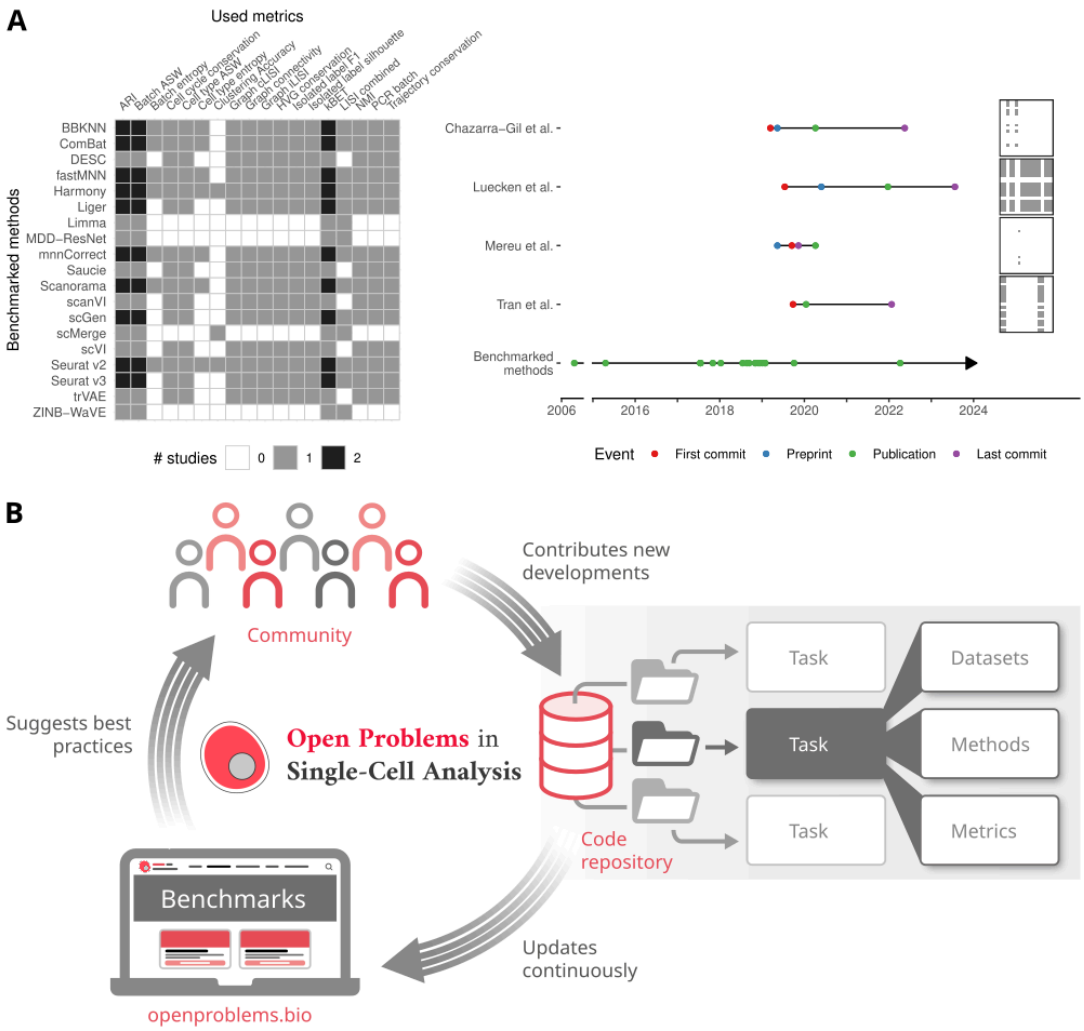


Figure 1: The Open Problems in Single-cell Analysis living benchmarking platform. A) Overview and timeline of published benchmarks of single-cell batch integration. Four publications have benchmarked 19 methods using 18 metrics. Light grey and black squares indicate whether one or two benchmarks include this method-metric combination (left). Arrows indicate the range of publication times of methods included in the benchmark. B) Schematic diagram of the Open Problems platform. The Open Problems platform consists of tasks that are broken down into datasets, methods, and metrics. The community contributes code to these tasks in the platform, which uses these contributions to extend the benchmarks that are run and pushed to the Open Problems website. The community can then consult the website for guidance on method selection.

To achieve this goal, we developed the Open Problems in Single-Cell Analysis (Open Problems) platform. The Open Problems platform is an open-source, extensible, living benchmarking framework that enables quantitative evaluation of best practices in single-cell analysis. It combines a permissively licensed GitHub repository (https://github.com/openproblems-bio/openproblems) with community-defined tasks, an automated benchmarking workflow, and a website to explore the results. Currently, Open Problems includes 12 defined tasks, in which 81 datasets are used to evaluate 171 methods

using 37 metrics. These tasks were defined by community engagement, including on the public GitHub repository, in weekly community meetings, and at a hackathon in March 2021 with over 50 participants. This broad involvement has already led to new benchmarking insights and best practice recommendations, while improving and standardizing previously published benchmarks. We envision that Open Problems' community-defined standards for progress in single-cell data science will raise the bar for the selection and evaluation of methods, provide targets for novel method innovation, and enable developers without single-cell expertise to contribute to the field.

To enable truly living benchmarks, we designed a standardised and automated infrastructure that allows members of the single-cell community to contribute to Open Problems in a seamless manner (**Methods**). Each Open Problems task consists of datasets, methods, and metrics (**Fig 1b**). Datasets define both the input and the ground truth for a task, methods attempt to solve the task, and metrics evaluate the success of a method on a given dataset. We provide cloud infrastructure to enable centralised benchmarking when new methods, datasets, or metrics are added to our platform. Within each task, every method is evaluated on every dataset using every metric, and each method is then ranked on a per-dataset basis by the average normalised metric score and presented in a summary table on the Open Problems website (https://openproblems.bio). Normalisation is used to make metric ranges comparable for comparison and visualization of method results without impacting the metric's ability to highlight method outliers (**Methods**).

Community engagement on the platform is centred around an open discussion forum, open code contribution opportunities, and task leadership. Task leaders are community members who have contributed substantially to a task, assume organisational responsibilities for the task, and are ultimately responsible for task definition, maintenance, and facilitation of community contributions. Task definitions, choices of metrics, and implementations of methods are discussed on our GitHub repository and can be easily amended by pull requests which are reviewed by task leaders and the core infrastructure team. While this community-centered approach may lead to, for example, suboptimal metrics being contributed, it also facilitates a self-cleansing process whereby metrics can be removed or amended if limitations or biases are uncovered. In this manner Open Problems promotes the longevity of hosted benchmarks.

To enable seamless community involvement in Open Problems, we have designed our platform to leverage cloud infrastructure that provides reproducibility, accessibility, and automation (**Supplementary Figure 1**). Each task is organised as a directory with subdirectories for datasets, methods, metrics, and utilities. Each task must contain at least one dataset, one metric, and two baseline methods, which provide upper and lower bounds for performance of the task. Components (i.e. dataset loader, method, or metric) are Viash components which exist as a single script (implemented in Bash, Python, or R) and some metadata (a YAML file named config.vsh.yaml) in the relevant subdirectory[27]. Adding a new method is as simple as opening a pull request to the repository and adding a new file that follows the API for that task. When a community member adds a component, the new contribution is automatically tested in the cloud. When all tests pass and the new contribution is accepted, the results from the new

contribution are automatically submitted to the Open Problems website. To maximise reproducibility, each component is run within a versioned Docker container defined by the method contributor, and all data is downloaded from public repositories, including Figshare, the Gene Expression Omnibus (GEO)[28], and CELLxGENE[29].

Building on previous work defining open challenges in single-cell analysis[30] and independent benchmarking studies in single-cell genomics[20,21,23,31–38], we started by defining nine Open Problems tasks (**Fig. 2a**), which extends to 12 with the inclusion of subtasks. While several tasks were directly informed by published benchmarking papers (e.g., batch correction[20], cell-cell communication[39]), others were defined by method developers in the single-cell community (e.g., spatial decomposition). The Open Problems platform was seeded with these tasks as they represent a cross-section of important and (mostly) well-researched tasks in single-cell genomics. We envision for this set of tasks to be a starting point for further community development to address and refine further open problems in single-cell analysis.

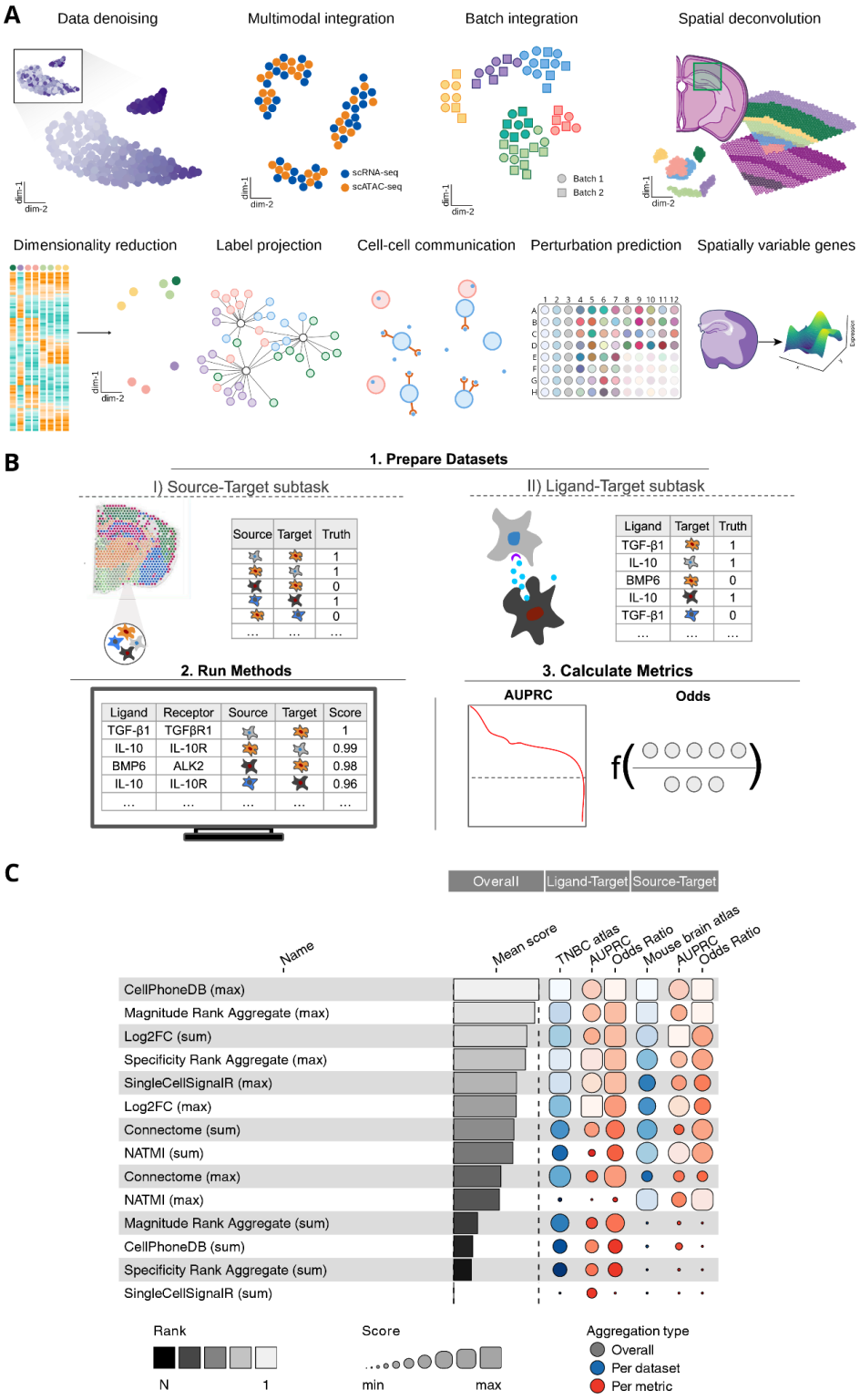


Figure 2: Task overview, setup and results. A) Overview of the nine tasks currently included in the Open Problems platform. Batch integration and cell-cell communication (CCC) consist of three and two subtasks respectively, making up the current total of 12 tasks. B) Schematic diagram of the CCC task. This task includes two subtasks defined by different types of ground truth: spatial cell type co-localization

in the source-target subtask and cytokine profiling in the ligand-target subtask. Methods are run on each subtask to score the likelihood of interaction between source and target cell types or ligand and target cell types. Finally, the area under the precision-recall curve (AUPRC) and the odds ratio of true to false positive interactions in the top 5% of predicted pairs are used to score method outputs (**Supplementary Note 1.1**). C) Collated results of both CCC subtasks. Methods are ranked using the mean of the overall score for each subtask (shown as "TNBC Atlas" and "Mouse brain atlas" blue boxes respectively). These overall scores are computed as the mean of all scaled metric results (red boxes). Linear scaling is performed using random and perfect baseline methods, whose performance is set to 0 and 1, respectively (see Methods). The results shown here are from Open Problems v1[40].

A typical task setup can be exemplified by the cell-cell communication (CCC) task (**Fig 2b; Supplementary Note 1.1)**. The goal of cell-cell communication inference methods is to infer which cell types are communicating within a tissue to mediate tissue function. Typical algorithms base predictions on the expression of ligand and receptor genes in dissociated single-cell data[41]. Ground-truth data for cellular communication are challenging to obtain. Thus, this task is divided into two subtasks that use different proxies for this ground truth: spatial colocalization (source-target subtask) and cytokine activity (ligand-target subtask). As the CCC methods included in this task[42–45] typically score ligand-receptor pairs using either their expression magnitude or cell-type specificity, *mean* and *max* aggregation functions are used to score interaction strengths between source and target cell types (source-target task) or ligands and target cell types (ligand-target task)[42–45]. The outputs of these methods are finally evaluated using the area under the precision-recall curve and odds ratios. These metrics measure how well ground truth source-target (co-localized cell types) or ligand-target (cytokine activity within a cell type) pairs are prioritised when ranking all interactions and how many true pairs are found in the top 5%, respectively.

While the CCC task was contributed to Open Problems on the basis of a published benchmark[39], the task definition and metrics evolved based on input from the community and the Open Problems team. This process has enabled the Open Problems results to generate insight beyond the initial publication (**Fig 2c**), which focused predominantly on the comparison of CCC databases and showed variable method performance across tasks. In the CCC Open Problems task, we find that methods that rely on expression magnitude outperform approaches that rely on expression specificity. Indeed, the top performers across tasks are CellPhoneDB and LIANA's ensemble model of expression magnitude scoring methods. Furthermore, *max* aggregation of ligand-receptor scores outperformed *mean* aggregation across tasks and methods. This improved inference of cellular communication using only the top-predicted interactions suggests that methods are better at prioritising a small fraction of relevant interactions while being prone to noise when their full interaction rankings are considered. Thus, analysts interpreting CCC results may likewise want to focus only on the most high-scoring predictions when inferring which cell types interact (**Supplementary Note 1.1**).

Using this combination of expert knowledge with community input, in this manuscript we also provide best-practice recommendations for preprocessing and method selection for label projection, dimensionality reduction for 2D visualisation, batch integration, spatial

decomposition, denoising, and matching cellular profiles across modalities (**Supplementary Note 1**). For example, on all four reference datasets currently included in the Open Problems label projection task, a simple logistic regression model outperforms more complex methods that explicitly model batch effects, such as Seurat[46] or scANVI[47], even when noise is added to the training data (**Supplementary Note 1.2**). Moreover, we also show that it is easier to correct for batch effects in single-cell graphs compared to in latent embeddings or expression matrices (**Supplementary Note 1.4**), that denoising methods perform best with non-standard preprocessing approaches that better stabilise variance (**Supplementary Note 1.6**), and that simple models tend to outperform more complex ones for perturbation prediction (**Supplementary Note 1.8**). Overall, Open Problems tasks are continuously updated benchmarks that increase in robustness as new methods are developed and more complex datasets become available. Our vision is that these benchmarks will form the basis for best-practice recommendations by groups such as Single-cell Best Practices (https://www.sc-best-practices.org/).

Open Problems living benchmarking tasks also function as a quantifiable target for the development of new methods. This problem definition is particularly useful for the wider machine learning community that may lack domain knowledge (i.e. single-cell expertise). Leveraging the batch integration and matching modality tasks as a basis, we previously set up popular competitions for multimodal data integration at NeurIPS 2021[48,49] and 2022, with over 260 and 1,600 participants, respectively. In these competitions, the developers of multiple top performers had no prior experience with single-cell data, yet were able to submit solutions that substantially outperform state-of-the-art methods[48]. We envision that the Open Problems platform will drive method development by improving the accessibility of open challenges in single-cell analysis via defined tasks. To promote this, Open Problems enables method developers to submit both prototype and final solutions to the platform for automated evaluation against the current state-of-the-art. Open Problems results, which are made available under a Creative Commons Attribution licence (CC-BY), can then be included in the respective method papers. Similarly, entirely new benchmarks can be implemented as tasks, run via Open Problems, and published separately while remaining updatable.

Taken together, the Open Problems platform is a community resource that quantitatively defines open challenges in single-cell analysis, determines the current state-of-the-art solutions, promotes method development to improve on these solutions, and monitors progress towards these goals. Open Problems addresses issues observed in bespoke and decentralised benchmarking by providing standardised but flexible infrastructure and task definitions. Thereby, Open Problems enables broader accessibility for scientists to contribute to the advancement of the field of single-cell analysis. We envision Open Problems to bring about a shift in perspective on method selection for data analysts and method evaluation for developers, supporting a transition towards higher standards for methods in single-cell data science.

Online Methods

Infrastructure

The Open Problems infrastructure prioritises automation, reproducibility, and ease of contribution. Where possible, all steps involved in the integration of new contributions to the living benchmark are automated with minimal manual review. All of the components involved in generating the benchmark are publicly accessible and documented, and contributing guides are made available to ensure that all community members are able to contribute to the benchmark. Briefly, the Open Problems infrastructure consists of two GitHub repositories that orchestrate continuous integration and continuous deployment via GitHub Actions workflows, using Viash[27], Nextflow[50] and AWS Batch to run the benchmark, and Quarto and Netlify[51] to render and host the website (**Fig S1**).

Code structure

Each task in the benchmark is broken down into three core components: datasets, methods, and metrics. Datasets provide a single-cell dataset with known ground truth corresponding to the task, methods perform the task, and metrics evaluate the methods' performance with respect to the defined task (**Fig 1b**). Each time the living benchmark is updated, every method is run on all datasets and evaluated using all metrics in the task (or subtask) to give the final score presented on the website. In practice, not all combinations of methods and metrics execute successfully (e.g. some methods do not scale to large datasets). In these cases, we track the causes of unsuccessful evaluations, such as out of memory or execution errors and list them both as a pie-chart visualisation on the benchmark summary table and provide more detail on the error in a "Quality control results" section of the results page.

Dataset loaders, methods, and metrics are written as Viash components[27], which consist of a script (Python or R) combined with metadata to generate standalone executables and modular Nextflow modules. Datasets return an AnnData object[52], methods accept this AnnData object and return a modified AnnData object, and metrics accept the modified AnnData object and return a floating-point value. Additionally, the Open Problems repository also provides a number of utility functions used across multiple tasks. These include data loaders, which download publicly available data that may be used as datasets in multiple tasks, normalisers, which provide standardised approaches to normalising raw data, and Docker images, which provide common sets of dependencies used across many datasets, methods, and metrics.

Metric normalisation

Metrics can have different effective ranges when evaluating methods for a particular task. While these different ranges may not affect method comparisons using only one metric, they do affect benchmarking results when multiple performance metric scores must be combined to give an overall ranking of methods (e.g., for visualisation). In order to avoid different metric ranges affecting the contribution of each metric to the final score and therefore the ranking for visualization, we normalise metric outputs to make them comparable. Specifically, we use a variant of min-max scaling that is robust to outliers by relying on a system of "baseline methods"

that determine the effective range of a metric. These baseline methods are designed to approximate both optimal and random performance on a given task for each metric. Since metrics in a task may be optimised by different baseline methods, we consider the optimum score of a given metric as the maximum score achieved by any baseline method and random performance as the minimum score achieved by any baseline method. All method scores are then normalised to this range such that optimum performance corresponds to a normalised score of 1 and random performance to a normalised score of 0. We specifically avoid transforming metric distributions within these ranges to a common distribution as the underlying methods may perform differently across different facets of a task. Indeed, methods appearing as outliers within a metric's pre-specified range should be highlighted by this metric by design. Following best practices for machine learning competitions[53], each method's score is then averaged over all normalised scores to give the method's overall score. Note that for some metrics (e.g., R squared in a regression task), it is possible to perform arbitrarily worse than random. In this case, methods may achieve scores significantly less than 0.

Benchmark procedure

The Open Problems living benchmark is run every two months on all contributions via AWS Batch, or in parts whenever triggered by a task leader due to a relevant update of datasets, methods, or metrics. A Nextflow workflow collects all relevant datasets, runs all methods on each dataset, and computes all metrics on each method-dataset pair for a given task. The results of this benchmark are metric scores and compute resources used for each method on each dataset. The computation for this pipeline is run on AWS Batch and stored on AWS S3. Following the successful completion of a benchmark run, Nextflow Tower triggers a GitHub webhook to download the results from S3, process them, and commit them to the Open Problems website repository, which displays these results on the website. All components of our Nextflow pipeline are made fully available on github and documented on our website (https://openproblems.bio/documentation), enabling the benchmark to also be run locally.

Continuous Integration

To ensure community contributions to Open Problems function as intended, we implement a series of automated tests applied to all contributions to the repository. Unit tests ensure that all tasks, datasets, methods, and metrics metadata and outputs to the expected API. This is achieved through a combination of universal- and task-specific API checks, which confirm that each function produces the intended output defined in each task. Additionally, each task must define a sample dataset and method, which are used as input for testing the implementation of methods and metrics respectively.

Continuous Deployment

The Open Problems website (https://openproblems.bio) is a Quarto website (https://quarto.org) composed of a) static content stored in the website repository, b) metadata and results data from the latest benchmarking run, and c) the documentation which is generated by rendering Quarto Notebooks using the code that is being documented. When a new benchmarking run of a task has completed, a pull request is created to commit the updated results to the website

repository. The results are visualised as a "funkyheatmap" with filters to allow users to explore the results interactively[54].

Development

Ease of contribution by the community is one of the central design principles for the Open Problems infrastructure. To facilitate the contribution of new methods and optimization of existing methods, we provide a command-line interface (CLI) to Open Problems. This CLI enables developers to locally evaluate the results of their contributions in a targeted manner (i.e. running only the submitted method rather than the full benchmark) and without prior experience with the Open Problems repository. The CLI provides a simple one-line command to load any dataset, run any method (given a dataset), or compute a metric (given the output of a method). Additionally, a detailed guide for contributing datasets, methods, metrics, and new tasks is maintained at https://github.com/openproblems-bio/openproblems/blob/main/CONTRIBUTING.md.

Open Problems tasks

Open Problems tasks are classified as stub or full tasks to denote task maturity. Stubs consist of at least one dataset, three methods, and one metric while tasks are regarded as full once they encompass at least two datasets, six methods, and a metric. Tasks that do not qualify as stubs are regarded as "*under discussion*" and are omitted here. This classification serves to communicate to users at which point meaningful guidance can be derived from the results of an Open Problems benchmark. Here we outline the setup of currently defined tasks in the Open Problems platform, encompassing nine full tasks (including five subtasks) and one stub task. Details on datasets, methods, and metrics, as well as discussion of task results and interpretation, are elaborated on in **Supplementary Note 1**.

Cell-cell communication

To harmonise the different tools and resources, we used the LIANA framework as a foundation for the cell-cell communication task[39]. To generate a ground truth for CCC benchmarking, we used alternative data modalities that provide insight into cellular communication such as spatial proximity and cytokine signalling. Each modality corresponds to a subtask. In the source-target subtask, we assess whether putatively interacting cell types are close to each other in spatial data. In the ligand-target subtask, downstream cytokine activities are used to infer whether a cytokine ligand was indeed active within a target cell type.

Label projection

To benchmark label projection methods, each dataset is divided into reference and query subsets. Methods are trained on the reference data subset and predict cell type labels for the query data subset. This prediction is evaluated against the true labels to quantify method performance. Different train and test splits are evaluated on several datasets.

Dimensionality reduction for 2D visualisation

The dimensionality reduction task attempts to quantify the ability of methods to embed the information present in complex single-cell studies into a two-dimensional space. Thus, this task is specifically designed for dimensionality reduction for visualisation and does not consider other uses of dimensionality reduction in standard single-cell workflows such as improving the signal-to-noise ratio (and in fact several of the methods use PCA as a pre-processing step for this reason). In this task, methods must take gene expression data as input and produce a two-dimensional coordinate for each cell. Unlike most tasks, there is a strongly suggested normalization approach for methods in the dimensionality reduction task that is passed to each method alongside the raw count data (normalisation to 10,000 counts per cell and log transformed (log-10k)). Pre-normalised matrices are required to enforce consistency between the metric evaluation (that typically uses pre-normalized expression data) and the method runs. When these are not consistent, methods that use the same normalisation as used in the metric tend to score more highly. For some methods we also evaluate the pre-processing recommended by the method.

Batch integration

In this task, we evaluate batch integration methods on their ability to remove batch effects in the data while conserving variation attributed to biological effects. As methods that integrate batches can output three different data formats (feature matrices, embeddings and/or neighbourhood graphs), we split the batch integration task into three subtasks. As input, all tasks take a combined normalised dataset with multiple batches and consistent cell-type labels. The respective batch-integrated representation (matrix, embedding, or graph) is then evaluated using sets of metrics that capture how well batch effects are removed and whether biological variance is conserved. We have based this particular task on a recent, extensive benchmark of single-cell data integration methods[20].

Spatial decomposition

The spatial decomposition task revolves around inferring relative cell type abundances in array-based spatial transcriptomics data. Specifically, the task requires methods to estimate the composition of cell identities (i.e., cell type or state) that are present at each capture location (i.e., spot or bead). The cell identity estimates are presented as proportion values, representing the proportion of the cells at each capture location that belong to a given cell identity. The faithfulness of this inference is evaluated using several metrics. In this task, we distinguish between reference-based decomposition and de novo decomposition, where the former leverages external data (e.g., scRNA-seq or scNuc-seq) to guide the inference process, while the latter only works with the spatial data. In this task, it is required that all datasets have an associated reference single-cell data set to perform reference-based decomposition, but methods are free to ignore this information to perform de novo decomposition instead. All methods benchmarked so far require a scRNA-seq reference to learn the cell-type-specific transcriptomics signature.

Denoising

Single-cell RNA-sequencing data can be notoriously noisy, with molecular capture rates that often hover around 40% for droplet-based sequencing[55] and up to 95% of measured zeros[56]. To address this noise, data augmentations that denoise or “impute” scRNA-seq expression matrices have been proposed. The data denoising task attempts to evaluate the major data denoising tools and to implement reasonable and universal metrics across a variety of datasets. The methods that are considered take as input a scRNA-seq expression matrix, which is then randomly partitioned into “train” and “test” subsets using the molecular cross validation (MCV) approach[57]. MCV creates train and test splits by simulating two random samples from the observed reads in each cell of the dataset. Once the training set has been denoised, its similarity to the testing set is assessed via one of several loss functions. Although datasets are assumed to already contain only the cells and genes that pass initial pre-processing steps, further normalisation is considered a part of the evaluated method. To facilitate the comparison of model performance using MCV, each denoising method is applied to the “train” subset, and model outputs are evaluated against the “test” subset using various metrics.

Matching modalities

In this stub task, the goal is to learn a latent space where cells profiled by different technologies in different modalities are matched if they have the same state. We use jointly profiled data as ground truth so that we can evaluate when the observations from the same cell acquired using different modalities are similar. A perfect result has each of the paired observations sharing the same coordinates in the latent space. A method that can achieve this would be able to match datasets across modalities to enable multimodal cellular analysis from separately measured profiles.

Perturbation prediction

This task aims to single-cell drug perturbation prediction methods by creating a novel dataset of human peripheral blood mononuclear cells (PBMCs) treated with 144 different compounds. PBMCs from three healthy donors were treated with each compound for 24 hours, and single-cell gene expression profiles were measured. Compounds were selected based on diverse transcriptional signatures observed in CD34+ hematopoietic stem cells. In addition to the perturbation data, baseline single-cell RNA and chromatin accessibility measurements were collected from each donor using the 10x Multiome assay. This multi-omic data provides context for interpreting gene expression changes in response to perturbation and allows for the assessment of how different cell types respond to the same compound. This dataset was then used to evaluate the performance of various prediction methods in a Kaggle competition as part of the NeurIPS 2023 competition track.

Spatially variable genes

Spatially variable genes (SVGs) are genes whose expression levels vary significantly across different spatial regions within a tissue or across cells in a spatially structured context. This benchmark evaluates various methods for detecting SVGs using realistic simulated datasets derived from real-world spatial transcriptomics data. Synthetic data is generated by mixing a

Gaussian Process (GP) model and a non-spatial model to generate gene expressions with various spatial variability.

Code and data availability

All Open Problems code is publicly available at https://www.github.com/openproblems-bio/openproblems. This code includes data loaders for all datasets used with associated metadata on where this data came from. Code to reproduce the figures is publicly available at https://github.com/openproblems-bio/nbt2023-manuscript. Furthermore, detailed information on all datasets are available at https://openproblems.bio/datasets. Documentation for the platform and contribution guides can be found at https://openproblems.bio/documentation.

Author contributions

M.D.L., S.G., and D.B.B. conceptualized the idea. M.D.L., S.G., D.B.B., R.C., and O.B.B. developed the infrastructure. M.D.L., S.G., D.B.B., R.C., D.C.S., N.S.M., L.Z., G.P., W.L., D.D., M.E.V., M.F.M., A.A., E.D., Q.Q., A.S., A.B., and Z.L. formalized a benchmarking task. M.D.L., S.G., D.B.B., R.C., D.C.S., N.S.M., L.Z., G.P., W.L., D.D., M.E.V., D.S.M., M.F.M., A.A., E.D., Q.Q., D.J.O., M.K., O.B.B., K.W., S.N.Y., A.S., A.B., Z.L., C.A-E., E.d.V.B., A.T.C., B.D., C.E., V.K., H.S., V.S., and A.T. contributed to the codebase. M.D.L., S.G., R.C., D.C.S., N.S.M., L.Z., G.P., W.L., D.D., L.D. and K.W. analysed the results. M.D.L., S.G., D.B.B., J.M.B., A.O.P., J.S-R., D.W., L.P., Y.S., F.J.T., S.K. provided resources and supervised the work. M.D.L., S.G., D.B.B., R.C., D.C.S., N.S.M., L.Z., G.P., W.L., D.D. coordinated the research. M.D.L., S.G., D.B.B., F.J.T., and S.K. acquired funding for the work. M.D.L., S.G., D.B.B., R.C., D.C.S., N.S.M., L.Z., G.P., W.L., D.D., M.E.V., M.F.M., A.A., E.D., Q.Q., D.J.O., M.K., O.B.B., A.S., A.B., Z.L., B.R., J.M.B., A.O.P., C.A-E., E.d.V.B., A.B., C.B.G-B., A.T.C., B.D., C.E., S.F., A.G., S.H., Y.J., V.K., G.L.M., M.G.L., R.L., D.R., H.S., V.S., A.T., G.X., and C.X. contributed to benchmarking task definition. M.D.L., S.G., D.B.B., R.C., D.C.S., N.S.M., L.Z., G.P., W.L., D.D., M.E.V., and D.S.M. prepared the manuscript. All authors reviewed the manuscript.

Group author: The Open Problems Jamboree Members
Bastian Rieck, Constantin Ahlmann-Eltze, Eduardo da Veiga Beltrame, Carmen Bravo González-Blas, Ann T Chen, Benjamin DeMeo, Can Ergen, Swann Floc'hlay, Adam Gayoso, Stephanie Hicks, Yuge Ji, Vitalii Kleshchevnikov, Gioele La Manno, Maximilian G. Lombardo, Romain Lopez, Dario Righelli, Hirak Sarkar, Valentine Svensson, Alexander Tong, Galen Xing, Chenling Xu

Conflict of interest Statement

M.D.L. consults for CatalYm GmbH, contracted for the Chan Zuckerberg Initiative and received speaker fees from Pfizer and Janssen Pharmaceuticals. S.G. has equity interest in Immunai Inc. D.B.B. is a paid employee of and has equity interest in Cellarity Inc. R.C. has equity interest in Data Intuitive BV. L.Z. has consulted for Lamin Labs GmbH. W.L. contracted for Protein

Evolution Incorporated. From 2019 to 2022 A.A. was a consultant for 10X Genomics. From October 2023 E.D. has been a consultant for EnsoCell Therapeutics. O.B.B is currently an employee of Bridge Bio Pharma. A.S. consults for Cellarity Inc. and Exvivo Labs Inc. A.B. is a paid employee of and has equity interest in Cellarity, Inc. J.B. has equity interest in Cellarity, Inc. J.S.R. reports funding from GSK, Pfizer and Sanofi and fees/honoraria from Travere Therapeutics, Stadapharm, Astex, Owkin, Pfizer and Grunenthal. D.W. has equity interest in Immunai Inc. F.J.T. consults for Immunai Inc., Singularity Bio B.V., CytoReason Ltd, Cellarity, and has ownership interest in Dermagnostix GmbH and Cellarity. S.K. is a visiting professor at Meta and scientific advisor at Ascent Bio, Inc. E.d.V.B has ownership interest in Retro Biosciences and ImYoo Inc and is employed by ImYoo Inc. A.T.C. is an employee of Orion Medicines. B.D. is a paid employee of and has equity interest in Cellarity Inc. A.G. is currently an employee of Google DeepMind. Google DeepMind has not directed any aspect of this study nor exerts any commercial rights over the results. R.L. is an employee of Genentech. V.S. has ownership interest in Altos Labs and Vesalius Therapeutics. A.T. has an ownership interest in Dreamfold.

Acknowledgements

We received continuous support in numerous ways from Jonah Cool, Ivana Williams, and Fiona Griffin from the Chan Zuckerberg Initiative for this project, without whom we would not have come this far. We would also like to thank Mohammad Lotfollahi for early discussions on Open Problems.

This work was supported by the Chan Zuckerberg Initiative Foundation (CZIF; grant CZIF2022-007488 (Human Cell Atlas Data Ecosystem) and the Chan Zuckerberg Initiative DAF, an advised fund of Silicon Valley Community Foundation (grant number 2021- 235155) awarded to M.D.L., D.B.B., S.G., F.J.T., and S.K.. This work was co-funded by the European Union (ERC, DeepCell -101054957, to A.S. and F.J.T.). Views and opinions expressed are however those of the authors only and do not necessarily reflect those of the European Union or the European Research Council. Neither the European Union nor the granting authority can be held responsible for them. G.P. is supported by the Helmholtz Association under the joint research school Munich School for Data Science and by the Joachim Herz Foundation. Throughout this work, W.L. was supported by the National Institutes of Health under the Continuing Education Training Grants (T15). D.D. was supported by the European Union's Horizon 2020 research and innovation program (860329 Marie-Curie ITN "STRATEGY-CKD"). M.E.V. is supported by the National Institutes of Health (NIH) under the Ruth L. Kirschstein National Research Service Award (1F31CA257625) from the National Cancer Institute (NCI). E.D. is supported by Wellcome Sanger core funding (WT206194). This work was supported by the Research Foundation Flanders (FWO) [1SF3822N to L.D.]. B.R. is supported by the Bavarian state government with funds from the Hightech Agenda Bavaria. This research received funding from the Flemish Government under the "Onderzoeksprogramma Artificiele Intelligentie (AI) Vlaanderen" programme. E.V.B. Would like to thank the Caltech Bioengineering Graduate program and Paul W. Sternberg for support. C.B.G.-B. was supported by a PhD fellowship from Fonds Wetenschappelijk Onderzoek (FWO, 11F1519N). V.K. was supported by Wellcome Sanger core funding. G.L.M. received support from the Swiss National Science Foundation

(SNSF) grant PZ00P3_193445 and Chan Zuckerberg Initiative grants number 2022-249212 and 2019-002427. D.R. was supported by the National Cancer Institute of the National Institutes of Health (2U24CA180996).

References

1. Cao, J. *et al.* A human cell atlas of fetal gene expression. *Science* **370**, (2020).
2. Montoro, D. T. *et al.* A revised airway epithelial hierarchy includes CFTR-expressing ionocytes. *Nature* **560**, 319–324 (2018).
3. Plass, M. *et al.* Cell type atlas and lineage tree of a whole complex animal by single-cell transcriptomics. *Science* **360**, eaaq1723 (2018).
4. Zappia, L., Phipson, B. & Oshlack, A. Exploring the single-cell RNA-seq analysis landscape with the scRNA-tools database. *PLoS Comput. Biol.* **14**, e1006245 (2018).
5. Heumos, L. *et al.* Best practices for single-cell analysis across modalities. *Nat. Rev. Genet.* (2023).
6. Luecken, M. D. & Theis, F. J. Current best practices in single-cell RNA-seq analysis: a tutorial. *Mol. Syst. Biol.* **15**, e8746 (2019).
7. Donoho, D. 50 Years of Data Science. *Journal of Computational and Graphical Statistics* vol. 26 745–766 Preprint at https://doi.org/10.1080/10618600.2017.1384734 (2017).
8. Deng, J. *et al.* Imagenet: A large-scale hierarchical image database. in *2009 IEEE conference on computer vision and pattern recognition* 248–255 (Ieee, 2009).
9. Wang, A. *et al.* GLUE: A Multi-Task Benchmark and Analysis Platform for Natural Language Understanding. *arXiv [cs.CL]* (2018).
10. Kitano, H. *RoboCup-97: Robot Soccer World Cup I*. (Springer Science & Business Media, 1998).
11. Lanning, J. B. S. & Bennett, J. Netflix Prize. *Proc. KDD Cup and Workshop 2007*.
12. Moult, J. A decade of CASP: progress, bottlenecks and prognosis in protein structure prediction. *Curr. Opin. Struct. Biol.* **15**, 285–289 (2005).

13. Meyer, P. & Saez-Rodriguez, J. Advances in systems biology modeling: 10 years of crowdsourcing DREAM challenges. *Cell Syst* **12**, 636–653 (2021).
14. Sonrel, A. *et al.* Meta-analysis of (single-cell method) benchmarks reveals the need for extensibility and interoperability. *Genome Biol.* **24**, 119 (2023).
15. Brooks, T. G., Lahens, N. F., Mrčela, A. & Grant, G. R. Challenges and best practices in omics benchmarking. *Nat. Rev. Genet.* (2024) doi:10.1038/s41576-023-00679-6.
16. Buchka, S., Hapfelmeier, A., Gardner, P. P., Wilson, R. & Boulesteix, A.-L. On the optimistic performance evaluation of newly introduced bioinformatic methods. *Genome Biol.* **22**, 152 (2021).
17. Musgrave, K., Belongie, S. & Lim, S.-N. A Metric Learning Reality Check. *arXiv [cs.CV]* (2020).
18. Chambers, C. D. & Tzavella, L. The past, present and future of Registered Reports. *Nat Hum Behav* **6**, 29–42 (2022).
19. Saelens, W., Cannoodt, R., Todorov, H. & Saeys, Y. A comparison of single-cell trajectory inference methods. *Nat. Biotechnol.* **37**, 547–554 (2019).
20. Luecken, M. D. *et al.* Benchmarking atlas-level data integration in single-cell genomics. *Nat. Methods* **19**, 41–50 (2022).
21. Soneson, C. & Robinson, M. D. Bias, robustness and scalability in single-cell differential expression analysis. *Nat. Methods* **15**, 255–261 (2018).
22. Squair, J. W. *et al.* Confronting false discoveries in single-cell differential expression. *Nat. Commun.* **12**, 5692 (2021).
23. Chazarra-Gil, R., van Dongen, S., Kiselev, V. Y. & Hemberg, M. Flexible comparison of batch correction methods for single-cell RNA-seq using BatchBench. *Nucleic Acids Res.* **49**, e42 (2021).
24. Tran, H. T. N. *et al.* A benchmark of batch-effect correction methods for single-cell RNA sequencing data. *Genome Biol.* **21**, 12 (2020).

25. Mereu, E. *et al.* Benchmarking single-cell RNA-sequencing protocols for cell atlas projects. *Nat. Biotechnol.* **38**, 747–755 (2020).
26. Cao, Y. *et al.* The current landscape and emerging challenges of benchmarking single-cell methods. *bioRxiv* 2023.12.19.572303 (2023) doi:10.1101/2023.12.19.572303.
27. Cannoodt, R. *et al.* Viash: A meta-framework for building reusable workflow modules. *J. Open Source Softw.* **9**, 6089 (2024).
28. Edgar, R., Domrachev, M. & Lash, A. E. Gene Expression Omnibus: NCBI gene expression and hybridization array data repository. *Nucleic Acids Res.* **30**, 207–210 (2002).
29. Megill, C. *et al.* cellxgene: a performant, scalable exploration platform for high dimensional sparse matrices. *bioRxiv* 2021.04.05.438318 (2021) doi:10.1101/2021.04.05.438318.
30. Lähnemann, D. *et al.* Eleven grand challenges in single-cell data science. *Genome Biol.* **21**, 31 (2020).
31. Li, B. *et al.* Benchmarking spatial and single-cell transcriptomics integration methods for transcript distribution prediction and cell type deconvolution. *Nat. Methods* **19**, 662–670 (2022).
32. Hou, W., Ji, Z., Ji, H. & Hicks, S. C. A systematic evaluation of single-cell RNA-sequencing imputation methods. *Genome Biol.* **21**, 218 (2020).
33. Raimundo, F., Vallot, C. & Vert, J.-P. Tuning parameters of dimensionality reduction methods for single-cell RNA-seq analysis. *Genome Biol.* **21**, 212 (2020).
34. Sun, X., Lin, X., Li, Z. & Wu, H. A comprehensive comparison of supervised and unsupervised methods for cell type identification in single-cell RNA-seq. *Brief. Bioinform.* **23**, (2022).
35. Sun, S., Zhu, J., Ma, Y. & Zhou, X. Accuracy, robustness and scalability of dimensionality reduction methods for single-cell RNA-seq analysis. *Genome Biol.* **20**, 269 (2019).
36. Huang, Y. & Zhang, P. Evaluation of machine learning approaches for cell-type identification from single-cell transcriptomics data. *Brief. Bioinform.* **22**, (2021).

37. Avila Cobos, F., Alquicira-Hernandez, J., Powell, J. E., Mestdagh, P. & De Preter, K. Benchmarking of cell type deconvolution pipelines for transcriptomics data. *Nat. Commun.* **11**, 5650 (2020).
38. Cantini, L. *et al.* Benchmarking joint multi-omics dimensionality reduction approaches for the study of cancer. *Nat. Commun.* **12**, 124 (2021).
39. Dimitrov, D. *et al.* Comparison of methods and resources for cell-cell communication inference from single-cell RNA-Seq data. *Nat. Commun.* **13**, 3224 (2022).
40. Gigante, S. *et al. Openproblems-Bio/openproblems: v1.0.0*. (Zenodo, 2024). doi:10.5281/ZENODO.13769879.
41. Armingol, E., Baghdassarian, H. M. & Lewis, N. E. The diversification of methods for studying cell–cell interactions and communication. *Nat. Rev. Genet.* 1–20 (2024).
42. Efremova, M., Vento-Tormo, M., Teichmann, S. A. & Vento-Tormo, R. CellPhoneDB: inferring cell-cell communication from combined expression of multi-subunit ligand-receptor complexes. *Nat. Protoc.* **15**, 1484–1506 (2020).
43. Hou, R., Denisenko, E., Ong, H. T., Ramilowski, J. A. & Forrest, A. R. R. Predicting cell-to-cell communication networks using NATMI. *Nat. Commun.* **11**, 5011 (2020).
44. Raredon, M. S. B. *et al.* Computation and visualization of cell-cell signaling topologies in single-cell systems data using Connectome. *Sci. Rep.* **12**, 4187 (2022).
45. Cabello-Aguilar, S. *et al.* SingleCellSignalR: inference of intercellular networks from single-cell transcriptomics. *Nucleic Acids Res.* **48**, e55 (2020).
46. Stuart, T. *et al.* Comprehensive Integration of Single-Cell Data. *Cell* **177**, 1888–1902.e21 (2019).
47. Xu, C. *et al.* Probabilistic harmonization and annotation of single-cell transcriptomics data with deep generative models. *Mol. Syst. Biol.* **17**, e9620 (2021).
48. Lance, C. *et al.* Multimodal single cell data integration challenge: results and lessons learned. in *Proceedings of the NeurIPS 2021 Competitions and Demonstrations Track*

162–176 (2022).

49. Luecken, M. D. *et al.* A sandbox for prediction and integration of DNA, RNA, and proteins in single cells. in *Proceedings of the Neural Information Processing Systems Track on Datasets and Benchmarks 1 (NeurIPS Datasets and Benchmarks 2021)* (2021).
50. Di Tommaso, P. *et al.* Nextflow enables reproducible computational workflows. *Nat. Biotechnol.* **35**, 316–319 (2017).
51. Attardi, J. *Using Gatsby and Netlify CMS*. (Apress).
52. Virshup, I., Rybakov, S., Theis, F. J., Angerer, P. & Wolf, F. A. anndata: Access and store annotated data matrices. *Journal of Open Source Software* **9**, 4371 (2024).
53. Maier-Hein, L. *et al.* Why rankings of biomedical image analysis competitions should be interpreted with care. *Nat. Commun.* **9**, 5217 (2018).
54. Cannoodt, R., Deconinck, L. & Zappia, L. *Funkyheatmap/funkyheatmap: Funkyheatmap 0.5.1*. (Zenodo, 2025). doi:10.5281/ZENODO.14653648.
55. Hagemann-Jensen, M. *et al.* Single-cell RNA counting at allele and isoform resolution using Smart-seq3. *Nat. Biotechnol.* **38**, 708–714 (2020).
56. Hicks, S. C., Townes, F. W., Teng, M. & Irizarry, R. A. Missing data and technical variability in single-cell RNA-sequencing experiments. *Biostatistics* **19**, 562–578 (2018).
57. Batson, J., Royer, L. & Webber, J. Molecular Cross-Validation for Single-Cell RNA-seq. *bioRxiv* 786269 (2019) doi:10.1101/786269.

Supplementary Information: Defining and benchmarking open problems in single-cell analysis

Malte D. Luecken[1,2*], Scott Gigante[3*], Daniel B. Burkhardt[4*], Robrecht Cannoodt[5,6,7*], Daniel C. Strobl[1,8,9^], Nikolay S. Markov[10^], Luke Zappia[1,5,11^], Giovanni Palla[1,9^], Wesley Lewis[12^], Daniel Dimitrov[13^], Michael E. Vinyard[14,15,16^], D.S. Magruder[17^], Michaela F. Mueller[1,2,9^], Alma Andersson[18,19,20], Emma Dann[21], Qian Qin[15], Dominik J. Otto[22,23,24], Michal Klein[25], Olga Borisovna Botvinnik[26,27], Louise Deconinck[6,7], Kai Waldrant[5], Sai Nirmayi Yasa[5], Artur Szałata[1,11], Andrew Benz[4], Zhijian Li[15,16], The Open Problems Jamboree Members, Jonathan M. Bloom[28], Angela Oliveira Pisco[26,29], Julio Saez-Rodriguez[13], Drausin Wulsin[3], Luca Pinello[16], Yvan Saeys[6,7,30], Fabian J Theis[1,11,31 †], Smita Krishnaswamy[12,17,32 †]

Table of contents

Supplementary Methods

Infrastructure

The Open Problems infrastructure prioritizes automation, reproducibility, and ease of contribution. Where possible, all steps involved in the integration of new contributions to the living benchmark are automated with minimal manual review. All of the components involved in generating the benchmark are publicly accessible and documented, and contributing guides are made available to ensure that all community members are able to contribute to the benchmark. Briefly, the Open Problems infrastructure consists of two GitHub repositories that orchestrate continuous integration and continuous deployment via GitHub Actions workflows, using Viash[1], Nextflow[2] and AWS Batch to run the benchmark, and Quarto and Netlify[3] to render and host the website (**Fig S1**).

Code structure

Each task in the benchmark is broken down into three core components: datasets, methods, and metrics. Datasets provide a single-cell dataset with known ground truth corresponding to the task, methods perform the task, and metrics evaluate the methods' performance with respect to the defined task (**Fig 1b**). Each time the living benchmark is updated, every method is run on all datasets and evaluated using all metrics in the task (or subtask) to give the final score presented on the website. In practice, not all combinations of methods and metrics execute successfully (e.g. some methods do not scale to large datasets). In these cases, we track the causes of unsuccessful evaluations, such as out of memory or execution errors and list them both as a pie-chart visualization on the benchmark summary table and provide more detail on the error in a "Quality control results" section of the results page.

Dataset loaders, methods, and metrics are written as Viash components[1], which consist of a script (Python or R) combined with metadata to generate standalone executables and modular Nextflow modules. Datasets return an AnnData object[4], methods accept this AnnData object and return a modified AnnData object, and metrics accept the modified AnnData object and return a floating-point value. Additionally, the Open Problems repository also provides a number of utility functions used across multiple tasks. These include data loaders, which download publicly available data that may be used as datasets in multiple tasks, normalisers, which provide standardized approaches to normalising raw data, and Docker images, which provide common sets of dependencies used across many datasets, methods, and metrics.

Metric normalization

Metrics can have different effective ranges when evaluating methods for a particular task. While these different ranges may not affect method comparisons using only one metric, they do affect benchmarking results when multiple performance metric scores must be combined to give an overall ranking of methods (e.g., for visualization). In order to avoid different metric ranges

affecting the contribution of each metric to the final score and therefore the ranking for visualization, we normalize metric outputs to make them comparable. Specifically, we use a variant of min-max scaling that is robust to outliers by relying on a system of “baseline methods” that determine the effective range of a metric. These baseline methods are designed to approximate both optimal and random performance on a given task for each metric. Since metrics in a task may be optimized by different baseline methods, we consider the optimum score of a given metric as the maximum score achieved by any baseline method and random performance as the minimum score achieved by any baseline method. All method scores are then normalized to this range such that optimum performance corresponds to a normalized score of 1 and random performance to a normalized score of 0. We specifically avoid transforming metric distributions within these ranges to a common distribution as the underlying methods may perform differently across different facets of a task. Indeed, methods appearing as outliers within a metric’s pre-specified range should be highlighted by this metric by design. Following best practices for machine learning competitions[5], each method’s score is then averaged over all normalised scores to give the method’s overall score. Note that for some metrics (e.g., R squared in a regression task), it is possible to perform arbitrarily worse than random. In this case, methods may achieve scores significantly less than 0.

Benchmark procedure

The Open Problems living benchmark is run every two months on all contributions via AWS Batch, or in parts whenever triggered by a task leader due to a relevant update of datasets, methods, or metrics. A Nextflow workflow collects all relevant datasets, runs all methods on each dataset, and computes all metrics on each method-dataset pair for a given task. The results of this benchmark are metric scores and compute resources used for each method on each dataset. The computation for this pipeline is run on AWS Batch and stored on AWS S3. Following the successful completion of a benchmark run, Nextflow Tower triggers a GitHub webhook to download the results from S3, process them, and commit them to the Open Problems website repository, which displays these results on the website. All components of our Nextflow pipeline are made fully available on github and documented on our website (https://openproblems.bio/documentation), enabling the benchmark to also be run locally.

Continuous Integration

To ensure community contributions to Open Problems function as intended, we implement a series of automated tests applied to all contributions to the repository. Unit tests ensure that all tasks, datasets, methods, and metrics metadata and outputs to the expected API. This is achieved through a combination of universal- and task-specific API checks, which confirm that each function produces the intended output defined in each task. Additionally, each task must define a sample dataset and method, which are used as input for testing the implementation of methods and metrics respectively.

Continuous Deployment

The Open Problems website (https://openproblems.bio) is a Quarto website (https://quarto.org) composed of a) static content stored in the website repository, b) metadata and results data from the latest benchmarking run, and c) the documentation which is generated by rendering Quarto Notebooks using the code that is being documented. When a new benchmarking run of a task has completed, a pull request is created to commit the updated results to the website repository. The results are visualized as a "funkyheatmap" with filters to allow users to explore the results interactively[6].

Development

Ease of contribution by the community is one of the central design principles for the Open Problems infrastructure. To facilitate the contribution of new methods and optimization of existing methods, we provide a command-line interface (CLI) to Open Problems. This CLI enables developers to locally evaluate the results of their contributions in a targeted manner (i.e. running only the submitted method rather than the full benchmark) and without prior experience with the Open Problems repository. The CLI provides a simple one-line command to load any dataset, run any method (given a dataset), or compute a metric (given the output of a method). Additionally, a detailed guide for contributing datasets, methods, metrics, and new tasks is maintained at https://github.com/openproblems-bio/openproblems/blob/main/CONTRIBUTING.md.

Open Problems tasks

Open Problems tasks are classified as stub or full tasks to denote task maturity. Stubs consist of at least one dataset, three methods, and one metric while tasks are regarded as full once they encompass at least two datasets, six methods, and a metric. Tasks that do not qualify as stubs are regarded as "*under discussion*" and are omitted here. This classification serves to communicate to users at which point meaningful guidance can be derived from the results of an Open Problems benchmark. Here we outline the setup of currently defined tasks in the Open Problems platform, encompassing nine full tasks (including five subtasks) and one stub task. Details on datasets, methods, and metrics, as well as discussion of task results and interpretation, are elaborated on in **Supplementary Note 1**.

Cell-cell communication

To harmonize the different tools and resources, we used the LIANA framework as a foundation for the cell-cell communication task[7]. To generate a ground truth for CCC benchmarking, we used alternative data modalities that provide insight into cellular communication such as spatial proximity and cytokine signalling. Each modality corresponds to a subtask. In the source-target subtask, we assess whether putatively interacting cell types are close to each other in spatial data. In the ligand-target subtask, downstream cytokine activities are used to infer whether a cytokine ligand was indeed active within a target cell type.

Label projection

To benchmark label projection methods, each dataset is divided into reference and query subsets. Methods are trained on the reference data subset and predict cell type labels for the query data subset. This prediction is evaluated against the true labels to quantify method performance. Different train and test splits are evaluated on several datasets.

Dimensionality reduction for 2D visualization

The dimensionality reduction task attempts to quantify the ability of methods to embed the information present in complex single-cell studies into a two-dimensional space. Thus, this task is specifically designed for dimensionality reduction for visualization and does not consider other uses of dimensionality reduction in standard single-cell workflows such as improving the signal-to-noise ratio (and in fact several of the methods use PCA as a pre-processing step for this reason). In this task, methods must take gene expression data as input and produce a two-dimensional coordinate for each cell. Unlike most tasks, there is a strongly suggested normalization approach for methods in the dimensionality reduction task that is passed to each method alongside the raw count data (normalization to 10,000 counts per cell and log transformed (log-10k)). Pre-normalised matrices are required to enforce consistency between the metric evaluation (that typically uses pre-normalized expression data) and the method runs. When these are not consistent, methods that use the same normalization as used in the metric tend to score more highly. For some methods we also evaluate the pre-processing recommended by the method.

Batch integration

In this task, we evaluate batch integration methods on their ability to remove batch effects in the data while conserving variation attributed to biological effects. As methods that integrate batches can output three different data formats (feature matrices, embeddings and/or neighborhood graphs), we split the batch integration task into three subtasks. As input, all tasks take a combined normalized dataset with multiple batches and consistent cell-type labels. The respective batch-integrated representation (matrix, embedding, or graph) is then evaluated using sets of metrics that capture how well batch effects are removed and whether biological variance is conserved. We have based this particular task on a recent, extensive benchmark of single-cell data integration methods[8].

Spatial decomposition

The spatial decomposition task revolves around inferring relative cell type abundances in array-based spatial transcriptomics data. Specifically, the task requires methods to estimate the composition of cell identities (i.e., cell type or state) that are present at each capture location (i.e., spot or bead). The cell identity estimates are presented as proportion values, representing the proportion of the cells at each capture location that belong to a given cell identity. The faithfulness of this inference is evaluated using several metrics. In this task, we distinguish between reference-based decomposition and de novo decomposition, where the former leverages external data (e.g., scRNA-seq or scNuc-seq) to guide the inference process, while

the latter only works with the spatial data. In this task, it is required that all datasets have an associated reference single-cell data set to perform reference-based decomposition, but methods are free to ignore this information to perform de novo decomposition instead. All methods benchmarked so far require a scRNA-seq reference to learn the cell-type-specific transcriptomics signature.

Denoising

Single-cell RNA-sequencing data can be notoriously noisy, with molecular capture rates that often hover around 40% for droplet-based sequencing[9] and up to 95% of measured zeros[10]. To address this noise, data augmentations that denoise or “impute” scRNA-seq expression matrices have been proposed. The data denoising task attempts to evaluate the major data denoising tools and to implement reasonable and universal metrics across a variety of datasets. The methods that are considered take as input a scRNA-seq expression matrix, which is then randomly partitioned into “train” and “test” subsets using the molecular cross validation (MCV) approach[11]. MCV creates train and test splits by simulating two random samples from the observed reads in each cell of the dataset. Once the training set has been denoised, its similarity to the testing set is assessed via one of several loss functions. Although datasets are assumed to already contain only the cells and genes that pass initial pre-processing steps, further normalization is considered a part of the evaluated method. To facilitate the comparison of model performance using MCV, each denoising method is applied to the “train” subset, and model outputs are evaluated against the “test” subset using various metrics.

Matching modalities

In this stub task, the goal is to learn a latent space where cells profiled by different technologies in different modalities are matched if they have the same state. We use jointly profiled data as ground truth so that we can evaluate when the observations from the same cell acquired using different modalities are similar. A perfect result has each of the paired observations sharing the same coordinates in the latent space. A method that can achieve this would be able to match datasets across modalities to enable multimodal cellular analysis from separately measured profiles.

Perturbation prediction

This task aims to single-cell drug perturbation prediction methods by creating a novel dataset of human peripheral blood mononuclear cells (PBMCs) treated with 144 different compounds. PBMCs from three healthy donors were treated with each compound for 24 hours, and single-cell gene expression profiles were measured. Compounds were selected based on diverse transcriptional signatures observed in CD34+ hematopoietic stem cells. In addition to the perturbation data, baseline single-cell RNA and chromatin accessibility measurements were collected from each donor using the 10x Multiome assay. This multi-omic data provides context for interpreting gene expression changes in response to perturbation and allows for the assessment of how different cell types respond to the same compound. This dataset was then

used to evaluate the performance of various prediction methods in a Kaggle competition as part of the NeurIPS 2023 competition track.

Spatially variable genes

Spatially variable genes (SVGs) are genes whose expression levels vary significantly across different spatial regions within a tissue or across cells in a spatially structured context. This benchmark evaluates various methods for detecting SVGs using realistic simulated datasets derived from real-world spatial transcriptomics data. Synthetic data is generated by mixing a Gaussian Process (GP) model and a non-spatial model to generate gene expressions with various spatial variability.

Supplementary Note 1: Task definitions and results

Benchmarking results discussed below in Supplementary Notes 1.1-1.7 relate to Open Problems v1[12], while Supplementary Notes 1.8 and 1.9 relate to results from Open Problems v2.0.

Supplementary Note 1.1: Cell-cell communication

Motivation

Single-cell technologies are now routinely used to study cellular heterogeneity in tissues and organs. While these technologies typically measure each cell independently, tissue function is mediated by cellular communication, which is more challenging to measure in high throughput. This technological challenge has sparked the development of computational methods that infer cell-cell communication (CCC) patterns on the basis of dissociated cellular profiles, leading to an ever-growing number of computational tools developed for this purpose[13]. Different tools propose distinct preprocessing steps with diverse scoring functions that are challenging to compare and evaluate. Furthermore, each tool typically comes with its own set of prior knowledge. The challenges in evaluating the tools are further exacerbated by the lack of a gold standard to benchmark the performance of CCC methods.

Task description

To harmonize the different tools and resources, we used the LIANA framework as a foundation for the cell-cell communication task[7]. To generate a ground truth for CCC benchmarking, we used alternative data modalities that provide insight into cellular communication such as spatial proximity and cytokine signalling. Each modality corresponds to a subtask. In the source-target subtask, we assess whether putatively interacting cell types are close to each other in spatial data. In the ligand-target subtask, downstream cytokine activities are used to infer whether a cytokine ligand was indeed active within a target cell type.

Datasets

In the **Source-Target subtask**, cell type deconvolution proportions per spot from **10x Visium slides of adult mouse brain** were used to identify colocalized cell types. In particular, we used the SPOTlight[14] deconvolution method to spatially map the cell types present in an annotated murine adult brain scRNAseq dataset[15] onto sagittal adult mouse brain anterior and posterior slices 10x Visium slides (retrieved from Spatial Gene Expression v1 Chemistry datasets [https://tinyurl.com/10xVisiumDemonstration]). Then, Pearson correlation coefficients were calculated for each cell type pair and *z*-scaled to create a distribution of correlations, from which strongly correlated cell type pairs (*z*-score > = 1.645) were considered as colocalized, while the

remainder were considered as non-colocalized. In this subtask, colocalized and non-colocalized cell types were treated as the positive and negative class, respectively.

In the **Ligand-Target subtask**, we used downstream cytokine activities from an annotated **triple negative breast cancer dataset**[16] as assumed truth. Specifically, we inferred cytokine activities using CytoSig's high-quality signatures, together with the multivariate linear regression model ('mlm') method of decoupleR[17] for each cell type at the pseudobulk level. We considered cytokine signatures with positive coefficients and FDR-corrected p-value = < 0.05 in the target cell types as active, while the remainder were considered as inactive. In this subtask, active and inactive cytokines in the target cell types were treated as the positive and negative class, respectively.

Methods

Methods in the CCC task estimate interactions at the cell-type level, and hence consider ligand and receptor expression as their average expression per cell type. Each method uses ligand-receptor interaction scoring functions that can be subdivided into two classes: those that predominantly use the specificity of gene expression in one pair of cell types, and those that use the magnitude of gene expression in this cell type pair. If a method provided scoring functions of both classes, we used those as recommended by authors. Overall, five CCC methods were included in both subtasks, along with two ensemble approaches that aggregate the results from methods in the specificity and expression magnitude classes, respectively.

CellPhoneDBv2[18] is a method that calculates a mean of ligand-receptor expression as a measure of interaction magnitude, along with a permutation-based p-value as a measure of specificity. Here, we use the former to prioritize interactions, and subsequently filter these using a p-value threshold of 0.05.

logFC is a custom implemented method in LIANA, inspired by iTALK[19], that combines both expression and magnitude, and represents the average of one-versus-the-rest log2FC of ligand and receptor expression per cell type.

NATMI[20] uses the product of ligand-receptor expression as a measure of magnitude. As a measure of specificity, NATMI proposes the following:

$$specificity.edge = \frac{l}{l_s} \cdot \frac{r}{r_s},$$

where *l* and *r* represent the average expression of ligand and receptor per cell type, and l_s and r_s represent the sums of the average ligand and receptor expression across all cell types. We use its specificity measure, as recommended by the authors for single-context predictions.

Connectome[21] uses the product of ligand-receptor expression as a measure of magnitude, and the average of the z-transformed expression of ligand and receptor as a measure of specificity.

SingleCellSignalR[22] provides a magnitude score as $LRscore = \frac{\sqrt{l \cdot r}}{mean + \sqrt{l \cdot r}}$; where *l* and *r* are the average ligand and receptor expression per cell type, and *mean* is the mean of the expression matrix.

Specificity aggregate - The RobustRankAggregate[23] method was used to generate a consensus rank of specificity-focused scoring functions from the methods above.

Magnitude aggregate - The RobustRankAggregate[23] method was used to generate a consensus rank for the magnitude-focused scoring functions.

All above methods were run as re-implemented in LIANA (v0.1.9)[7] and used LIANA's consensus ligand-receptor database resource. This ensured that the methods themselves are comparable rather than just their respective databases. For all methods, we considered only interactions for which of both the ligand and receptor, and any of their subunits, were expressed in at least 10% of the source and target cell types. For heteromeric complexes, we considered the arithmetic mean expression of the members for all methods, except CellPhoneDB, for which we used the minimum member expression, as in the original implementation[18].

As a consequence of the partial 'ground' truth used in the benchmarks, the ligand-receptor predictions of the methods, currently implemented in the task, do not uniquely map to the assumed truth instances. Thus, aggregation of each methods' scores was carried out using both max and sum aggregation across predicted scores according to the assumed truth at hand. In the Source-Target subtask we aggregated the scores according to the source and target cell type in the assumed truth. In Ligand-Target subtask, we aggregated according to the ligands thought to be active in the target cell types.

Baseline Methods

Random Events serves as a negative control, where cell-cell communication events are randomly generated by random selection of ligand, receptor, source, target, and score.

True Events serves as a positive control, where the predicted data is a 1-to-1 copy of the solutions data.

Metrics

As both subtasks have a binary ground truth encoding, the same two metrics were used in both subtasks: the Odds ratio, and the area under the precision recall curve (AUPRC).

Odds ratio: The odds ratio represents the ratio of true and false positives within a set of prioritized interactions versus the same ratio for the remainder of the interactions. Thus, in this scenario odds ratios quantify the strength of association between the ability of methods to prioritize interactions and those interactions assigned to the positive class. Here we prioritize 0.05*N interactions, where N is the number of all assumed truth interactions. As odds ratios are

bound between 0 and +∞, with 1 signifying no association, they were sigmoid-transformed: $f(x) = 1 - 1/(1 + x/2)$; where *x* is the raw odds ratio.

AUPRC: a single number *[0-1]* that summarizes the area under the curve where *x* is the recall and *y* is the precision.

Task results

In both cell-cell communication subtasks, methods generally performed better than the random baseline, with magnitude-focused scoring functions generally outperforming those that focus on the specificity of the interactions across cell types (**Supplementary Figs 2 and 2**). Specifically, CellPhoneDB and LIANA's magnitude rank aggregate performed best in both subtasks.

In the Ligand-Target (cytokine activity) subtask, the methods performed better than random only when considering the odds ratio metric, while their PRAUCs were generally close to random. Nevertheless, when considering the odds ratios CellPhoneDB performed best, followed by both of LIANA's rank aggregates, SingleCellSignalR, log2FC, Connectome, and finally NATMI, with NATMI's prediction performance being close to random for both metrics.

In the Source-Target (spatial) subtask methods were notably better than random for both PRAUC and odds ratio metrics. CellPhoneDB and LIANA's magnitude rank performed best, followed by NATMI and log2FC. This implies that high-scoring interactions between ligand and receptor correctly capture anticipated communication events between adjacent cell types.

With the exception of the log2FC method, the max-aggregation of method scores typically performed better than their sum-aggregation counterparts. This suggests that few high scoring interactions between ligands and receptors more closely represent communication events between cells, while many low-scoring interactions do not. This notion is further supported by the relatively better performance of the methods when considering the odds ratio metric which focused on a small fraction of their predictions, contrasted to their weaker performance with regards to the PRAUC. Consequently, our results suggest that CCC inference methods are better at prioritising a small fraction of relevant interactions, while being prone to noise when their full interaction rankings are considered.

When evaluating the results of the CCC task, we must consider that the evaluations are based not on fixed measurements of true cellular interactions, but on approximations of biological reality which come with their own assumptions and limitations. Yet, the current subtasks enable the inclusion of more datasets and the extension of the benchmark setting with other modalities. To this end, we believe that despite their limitations the subtasks presented here provide a solid foundation that will facilitate the CCC method benchmark and development.

Supplementary Note 1.2: Label projection

Task motivation

Accurate identification of cell types and subtypes is necessary for interpretation of any single-cell dataset. Currently, as of 2022, most studies go through the time-consuming and subjective process of manual cell type annotation—leveraging the analyst's biological expertise to resolve cell types in the data. Automating this process would substantially speed up the analysis of single-cell data. A number of tools have recently been published that address this task. These tools enable accurate and automatic domain-specific cell type annotation by allowing to project cell type labels from high-quality reference dataset to another dataset. An additional benefit of such an approach is an increased power to detect rare cell types due to the use of auxiliary data. The label projection task in open problems aims to benchmark the performance of such methods.

Task description

To benchmark label projection methods, each dataset is divided into reference and query subsets. Methods are trained on the reference data subset and predict cell type labels for the query data subset. This prediction is evaluated against the true labels to quantify method performance. Different train and test splits are evaluated on several datasets.

Datasets

We included diverse datasets annotated at different levels of granularity to test performance across several species (human, mouse, zebrafish, *C. elegans*).

The **Pancreas** dataset is a collection of 6 datasets of transcriptomes from human pancreatic cells obtained with 6 different single-cell technologies: CelSeq, CelSeq2, Fluidigm C1, SMART-Seq2, inDrop and SMARTer. This dataset is collected and prepared in Luecken et al., *Nat Methods*, 2022, see github. We added a version of this dataset with 20% label noise. Dimensions: 16382 cells, 18771 genes. 14 cell types (avg. 1170±1703 cells per cell type).

Tabula Muris Senis Lung[24] includes all lung cells from Tabula Muris Senis, a 500k cell-atlas from 18 organs and tissues across the mouse lifespan. The lung cells are all collected via 10X sequencing, yielding 24540 cells and 16160 genes across 3 time points. Dimensions: 24540 cells, 17985 genes. 39 cell types (avg. 629±999 cells per cell type).

CeNGEN[25] is an atlas of the *C. elegans* nervous system, with neurons isolated by fluorescence-activated cell sorting from L4 stage larvae. Single-cell gene expression libraries

were prepared with 10x 3’ v3 chemistry and sequenced on Illumina NovaSeq6000. Dimensions: 100955 cells, 22469 genes. 169 cell types (avg. 597±800 cells per cell type).

Zebrafish[26] contains transcriptomes from several time points of zebrafish development during the first day. Dimensions: 26022 cells, 25258 genes. 24 cell types (avg. 1084±1156 cells per cell type).

Methods

We assessed standard classification methods (k-nearest neighbor classification, logistic regression, multilayer perceptron), as well as general algorithms (xgboost) and specific single-cell solutions (Seurat reference mapping, scANVI, scArches). Because k-nearest neighbor classification, logistic regression, multilayer perceptron and xgboost depend on data normalization, we included two versions of this step (see below). For scANVI and scArches we included two versions of the highly-variable gene selection step. As a simple test benchmark we also included the Majority vote method which predicts each cell to belong to the most abundant cell type.

Normalization methods

LogCP10k normalization is the most commonly used normalization method for single-cell data, that normalizes all cells to have 10,000 counts and then transforms each count with log(count+1).

Scran normalization[27] method estimates size factor for each cell relative to average cell size in the dataset, with a trick to reduce stochastic gene dropout contribution to these size factor estimates.

Classification methods

K-neighbors classifier[28] uses the "*k*-nearest neighbors" approach, which is a popular machine learning algorithm for classification and regression tasks. The assumption underlying KNN in this context is that cells with similar gene expression profiles tend to belong to the same cell type. For each unlabelled cell, this method computes the *k* labelled cells (in this case, 5) with the smallest distance in PCA space, and assigns that cell the most common cell type among its *k* nearest neighbors.

Logistic Regression[29] estimates parameters of a logistic function for multivariate classification tasks. Here, we use 100-dimensional centred and scaled PCA coordinates as independent variables, and the model minimizes the cross entropy loss over all cell type classes in the training data.

MLP[30] or "Multi-Layer Perceptron" is a type of artificial neural network that consists of multiple layers of interconnected neurons. Each neuron computes a weighted sum of all neurons in the previous layer and transforms it with nonlinear activation function. The output layer provides the final prediction, and network weights are updated by gradient descent to minimize the cross

entropy loss. Here, the input data is 100-dimensional centred and scaled PCA coordinates for each cell, and we use two hidden layers of 100 neurons each.

scANVI[31] or "single-cell ANnotation using Variational Inference" is a semi-supervised variant of the scVI[32] algorithm. Like scVI, scANVI is a variational autoencoder, which models cellular count data as a zero-inflated negative binomial distribution conditioned on cell batch, size factor and latent representation. Additionally to scVI, scANVI also leverages cell type labels in the generative modelling. In this approach, scANVI is used to predict cell type labels of the unlabelled test data from cell latent representations.

scArches+scANVI[33] or "Single-cell architecture surgery" is a deep learning method for mapping new datasets onto a pre-existing reference model, using transfer learning and parameter optimization. It first uses scANVI to build a reference model from the training data, and then freezes the reference model parameters and fine-tunes so-called adaptor weights that relate only to the test portion of the data. It then uses the latent cell representations to predict cell labels via scANVI.

Seurat reference mapping[34] is a cell type label transfer method provided by the Seurat package. Gene expression counts are first normalized by SCTransform before computing PCA. Then it finds mutual nearest neighbors, known as transfer anchors, between the labelled and unlabelled part of the data in PCA space, and computes each cell's distance to each of the anchor pairs. Finally, it uses the labelled anchors to predict cell types for unlabelled cells based on these distances.

XGBoost[35] is a gradient boosting decision tree model that learns multiple tree-based weak learners. Tree-predictions are aggregated in a weighted manner to produce a label prediction. Here, input features are normalized gene expression values.

Baseline Methods

Majority vote is a baseline-type method that predicts all cells to belong to the most abundant cell type in the dataset.

Random Labels serves as a negative control, where the labels are randomly predicted without training the data.

True Labels serves as a positive control, where the solution labels are copied 1 to 1 to the predicted data.

Metrics

Performance metrics include global accuracy, as well as weighted and unweighted F1 scores to balance prediction and recall over cell types and let the user assess cell type imbalance performance.

Accuracy measures the correct predictions in relation to the total predictions.

The **F1 score** measures the performances of the classification taking into account the precision and the recall of the model. This is preferred over the accuracy when the dataset is unbalanced. Here, we use a weighted average F1 score, where the F1 score for each cell type is weighted by the number of cells of this cell type.

Macro F1 score is a simple average of F1 scores per each of the cell types. It penalizes methods that make mistakes on cell types with few cells.

Task Results

Overall, logistic regression performed best across datasets. Specifically, logistic regression (log CP10k) had the best overall score in 5 out of 8 datasets, followed by Seurat in 2 datasets and MLP (log CP10k) in 1 dataset (**Supplementary Fig. 4**). While methods generally performed well on most datasets, the zebrafish (split by laboratory) dataset showed significantly worse performance across all methods, with Seurat performing best with a large relative lead on the other methods. Notably, this zebrafish dataset was the only dataset where training data did not contain all cell types. While this is a common real-life usage scenario for these methods, the performance was much worse than expected.

To interpret the results, we group label projection methods into 3 categories: standard machine learning (ML) methods, Seurat, and single-cell neural network (NN)-based methods. All standard ML methods (k-nearest neighbor classification, logistic regression, multilayer perceptron and xgboost) perform well with overall scores of over 0.7 per dataset. Only logistic regression (log scran) and MLP (log CP10k) on the pancreas dataset with added label noise (0.63 and 0.69 respectively) perform slightly worse. Seurat performed well with the second-best overall score averaged across datasets (0.79), while neural network-based single-cell methods (scANVI and scArches) performed less well with overall scores averaged across datasets of 0.46–0.63. Overall, standard ML methods and Seurat have stable high performance, including on the CenGEN dataset which has highly detailed annotations, making them an attractive first choice for the label projection task. NN-based methods appear to require more observations per cell type label to perform better, as indicated by their good performance (>0.78 overall score) on a simpler Pancreas dataset that contains the most cells per cell type label on average, and has the fewest number of cell types. Feeding NN models with all genes, rather than selecting 2000 highly-variable genes, led to improved performance on our datasets. In contrast, xgboost's third-best performance on the pancreas dataset could be explained by the smaller dataset size, suggesting that this method generalizes better with fewer cells than most others.

In this task, we additionally tested different preprocessing and noise perturbation strategies. The two normalization methods that we benchmarked do not seem to have a strong impact on the performance of the standard ML methods, except when additional noise was added. The addition of random label noise to the pancreas dataset mostly affected MLP and logistic regression (log scran), but not logistic regression (log CP10k), indicating the robustness of this normalization approach.

While this task includes many popular label projection methods and extends previously published benchmarks[36], it is not yet comprehensive. So far, we have only tested limited hyperparameter settings and preprocessing options, and are missing marker-based label projection methods such as Garnett[37]. Furthermore, the datasets in this task currently do not extend past 100 000 cells, while current single-cell atlases have more than 1 million cells. Compared with a previous benchmark[36], our task adds several important methods (Seurat, xgboost, MLP, scANVI), uses more diverse datasets, and includes several different preprocessing decisions. Future extension of this task by us and the larger community will include evaluations of unseen cell type label prediction, which have been found to distinguish label prediction method performance[36]. This will be addressed by using methods that can provide confidence for their cell type label prediction. We further plan to test method behavior with downsampled datasets and expand our panel of methods and datasets.

Supplementary Note 1.3: Dimensionality reduction for 2D visualization

Task motivation

Data visualization is an important part of all stages of single-cell analysis, from initial quality control to interpretation and presentation of final results. For bulk RNA-seq studies, linear dimensionality reduction techniques such as PCA and MDS are commonly used to visualize the variation between samples. While these methods are highly effective they can only be used to show the first few components of variation which cannot fully represent the increased complexity and number of observations in single-cell datasets. For this reason non-linear techniques (most notably t-SNE and UMAP) have become the standard for visualising single-cell studies. These methods attempt to compress a dataset into a two-dimensional space while attempting to capture as much of the variance between observations as possible. Many methods for solving this problem now exist. In general these methods try to preserve distances, while some additionally consider aspects such as density within the embedded space or conservation of continuous trajectories. Despite almost every single-cell study using one of these visualizations there has been debate as to whether they can effectively capture the variation in single-cell datasets[38].

Task description

The dimensionality reduction task attempts to quantify the ability of methods to embed the information present in complex single-cell studies into a two-dimensional space. Thus, this task is specifically designed for dimensionality reduction for visualization and does not consider other

uses of dimensionality reduction in standard single-cell workflows such as improving the signal-to-noise ratio (and in fact several of the methods use PCA as a pre-processing step for this reason). Unlike most tasks, methods for the dimensionality reduction task must accept a matrix containing expression values normalized to 10,000 counts per cell and log transformed (log-10k) and produce a two-dimensional coordinate for each cell. Pre-normalized matrices are required to enforce consistency between the metric evaluation (which generally requires normalized data) and the method runs. When these are not consistent, methods that use the same normalization as used in the metric tend to score more highly. For some methods we also evaluate the pre-processing recommended by the method.

Datasets

As this task is very general in nature we don't impose many constraints on the input dataset and in theory we could use almost any scRNA-seq dataset. We currently evaluate three datasets:

10x 5k PBMC includes peripheral blood mononuclear cells captured using 10x v3 chemistry produced as a reference dataset by 10x Genomics (retrieved from https://www.10xgenomics.com/resources/datasets/5-k-peripheral-blood-mononuclear-cells-pbmcs-from-a-healthy-donor-with-cell-surface-proteins-v-3-chemistry-3-1-standard-3-1-0). There are 20,822 features and 5,247 cells without any associated cell labels.

Olsson Mouse Blood 2016 is a 2016 SMART-seq dataset of mouse myeloid lineage differentiation containing 112,815 features for 660 cells with 4 cell type labels[39].

Nestorowa Mouse HSPC 2016 is a 2016 SMART-seq2 dataset of mouse hematopoietic stem cell differentiation containing 43,258 features for 1,920 cells with 3 cell type labels[40].

Zebrafish is a dataset of cells from zebrafish embryos throughout the first day of development, with and without a knockout of chordin, an important developmental gene[26]. There are 25,258 features for 26,022 cells from 24 cell types.

Methods

NeuralEE is a neural network implementation of elastic embedding. It is a non-linear method that preserves pairwise distances between data points[41]. NeuralEE uses a neural network to optimize an objective function that measures the difference between pairwise distances in the original high-dimensional space and the two-dimensional space. It is computed on both the recommended input from the package authors of 500 HVGs selected from a logged expression matrix (without sequencing depth scaling) and the default log-10k expression matrix with 1000 HVGs.

PCA or "Principal Component Analysis" is a linear method that decomposes a data matrix into orthogonal components that each maximize the captured variance[42]. The first two principal

components are chosen as the two-dimensional embedding. PCA is calculated on the log-10k expression matrix with and without selecting 1000 HVGs.

Diffusion maps use an affinity matrix to describe the similarity between data points, which is then transformed into a graph Laplacian[43]. The eigenvalue-weighted eigenvectors of the graph Laplacian are used to create the embedding. Diffusion maps are calculated on the log-10k expression matrix.

PHATE or “Potential of Heat-diffusion for Affinity-based Transition Embedding” uses the potential of heat diffusion to preserve trajectories in a dataset via a diffusion process[44]. It is an affinity-based method that creates an embedding by finding the dominant eigenvalues of a Markov transition matrix computed between cells in a single-cell k-nearest-neighbor graph using a smoothing graph kernel. We evaluate several variants including using the recommended square-root transformed counts per 10k matrix as input, this input with the gamma parameter set to zero and the normal log-10k transformed matrix with and without HVG selection.

PyMDE is a Python implementation of minimum-distortion embedding, a non-linear method that preserves distances between cells or neighborhoods in the high-dimensional space[45]. It is computed with options to preserve distances between cells or neighborhoods and with the log-10k matrix with and without HVG selection as input.

t-SNE or "t-distributed Stochastic Neighbor Embedding" converts similarities between data points to joint probabilities and tries to minimize the Kullback-Leibler divergence between the joint probabilities of the low-dimensional embedding and the high-dimensional data[46]. We use the implementation in the scanpy package which takes the log-10k expression matrix (with and without HVG selection), performs a PCA and supplies the results of the PCA to the t-SNE algorithm.

UMAP or "Uniform Manifold Approximation and Projection" is based on manifold learning techniques and topological data analysis to preserve the global and local structure of the data[47]. We perform UMAP on the log-10k expression matrix before and after HVG selection and with and without PCA as a pre-processing step.

densMAP is a modification of UMAP that adds an extra cost term in order to preserve information about the relative local density of the data[48]. It is performed on the same inputs as UMAP.

Baseline methods

Random coordinates: This method serves as a negative control, where the data is randomly embedded into a two-dimensional space, with no attempt to preserve the original structure.

Original input space: This serves as a positive control since the original high-dimensional data is retained as is, without any loss of information.

Metrics

The evaluation metrics for this task attempt to measure the preservation of either distances or local neighborhoods within the two-dimensional embedding compared to the original feature space.

Distance correlation evaluates the preservation of pairwise Euclidean distances between observations before and after dimensionality reduction by Spearman correlation[49].

Spectral distance correlation evaluates the preservation of pairwise diffusion distances[43] between observations before and after dimensionality reduction by Spearman correlation.

Trustworthiness focuses on the preservation of local neighborhoods by considering differences in the nearest neighbors of each observation before and after dimensionality reduction[50].

The **pyDRMetrics** package extends the trustworthiness approach by summarising nearest neighbor rankings that balance between local and global structure in different ways[51].

- **Continuity** measures the error of hard k-extrusions, that is points that are within the k-nearest neighbors before dimensionality reduction but outside the neighborhood afterward
- **co-KNN size** counts how many points are in both k-nearest neighbors before and after the dimensionality reduction
- **co-KNN AUC** is the area under the co-KNN curve
- The **local continuity** meta criterion is the co-KNN size with baseline removal which favors locality[52]
- The **local property** metric is a sum of the local co-KNN for values of k up to the value of k which maximizes the local continuity meta criterion
- The **global property** metric is a sum of the global co-KNN for values of k beyond that which maximizes the local continuity meta criterion

Density preservation is an alternative metric proposed by densMAP which measures whether the density of points around each observation is preserved, focusing less on exactly which points are nearby but instead whether a similar number of observations is found within a certain radius[48].

Task results

The majority of the methods in this task show similar performance when summarized across metrics and datasets (**Supplementary Fig. 5)**. The top performing method is densMAP (both with and without PCA as a pre-processing step). However it is important to note that most of the difference in performance between densMAP and other methods is due to the density preservation metric which this method specifically optimizes. In contrast, scores for more general metrics are similar across most methods. Furthermore, we observe that method variants which only consider the subset of highly variable genes generally perform slightly worse than

those that make use of all features, despite HVG selection being one of the most common steps in scRNA-seq analysis. t-SNE andPyMDE (the version that is optimized to preserve distances) are also top performers with PyMDE performing particularly well on the density correlation metric.

Overall these results emphasize the need to expand the scope of this task to better distinguish methods and capture the larger spectrum of available methods. Many more dimensionality reduction methods exist including modified versions of t-SNE with different distance matrices[53], Poincaire maps[54], self-assembling manifolds[55], hypersphere and hyperbolic embeddings[56], amalgams[57] and minimum-distortion embedding[45]. Other metrics have also been proposed such as the Jaccard distance metric[58]. Current metrics focus on technical aspects which, while important, may not measure how the visualization is used and interpreted. Further metrics that incorporate biological ground truth information such as cell type labels may better distinguish methods in terms of their usefulness to interpret biological results. To improve the robustness of this task, more datasets that cover a broader range of sequencing technologies and biological systems should be added.

Supplementary Note 1.4: Batch integration

Motivation

As single-cell technologies advance, single-cell datasets are growing both in size and complexity. Especially in consortia such as the Human Cell Atlas, individual studies combine data from multiple labs, each sequencing multiple individuals possibly with different technologies. This gives rise to complex batch effects in the data that must be computationally removed to perform a joint analysis. These batch integration methods must remove the batch effect while not removing relevant biological information. Currently, over 200 tools exist that aim to remove batch effects scRNA-seq datasets[59]. These methods balance the removal of batch effects with the conservation of nuanced biological information in different ways. This abundance of tools has complicated batch integration method choice, leading to several benchmarks on this topic[60–63]. Yet, benchmarks use different metrics, method implementations and datasets. Here we build a living benchmarking task for batch integration methods with the vision of improving the consistency of method evaluation.

Task Description

In this task we evaluate batch integration methods on their ability to remove batch effects in the data while conserving variation attributed to biological effects. As methods that integrate batches can output three different data formats (feature matrices, embeddings and/or neighborhood graphs), we split the batch integration task into three subtasks to account for this.

As input, all tasks take a combined normalized dataset with multiple batches and consistent cell type labels. The respective batch-integrated representation (matrix, embedding, or graph) is then evaluated using sets of metrics that capture how well batch effects are removed and whether biological variance is conserved. We have based this particular task on a recent, extensive benchmark of single-cell data integration methods[60].

Datasets

For this task, we used collections of datasets of differing complexity to get a good overview of the integration outputs. The curated dataset collections used here were sourced from the same recent benchmark that we base the task on[60]. Specifically, we used:

A **Pancreas** dataset that comprises of 6 datasets of transcriptomes from human pancreatic cells obtained with 6 different single-cell technologies: CelSeq, CelSeq2, Fluidigm C1, SMART-Seq2, inDrop and SMARTer. The dataset was processed as described in the published benchmark github (https://github.com/theislab/scib-reproducibility/tree/main/notebooks/data_preprocessing/pancreas). In total, it contains 16382 cells, 18771 genes. 14 cell types are annotated (avg. 1170±1703 cells per cell type).

The **Immune** dataset is a collection of 5 human immune cell datasets: one bone marrow dataset including 3 donors (Oetjen et al.) and four peripheral blood datasets (10X Genomics, Freytag et al., Sun et al. and Villani et al.). The dataset contains 33,506 cells, 8135 genes and 17 annotated cell types.

The **Lung** dataset[64] is a collection of 3 datasets from the same study, which were generated using Drop-seq and 10X chromium. We used the lung transplant and the lung biopsy data from 16 donors with 32,472 cells with 15,148 genes and 17 annotated cell types in total.

Methods

BBKNN or the batch balanced k nearest neighbors method, builds a joint kNN graph for each cell. First, a kNN graph is built within each defined batch separately, creating independent neighbor sets for each cell in each batch. These sets are then combined by forcing each cell to have a certain number of neighbors for each other batch. These connections are finally pruned to allow for batch-specific cell types[65]. BBKNN was run using the standard parametrization of 3 neighbors within each batch.

ComBat uses a linear mixed model that is fit using Empirical Bayes (EB) to correct for batch effects. It estimates linear and multiplicative batch parameters for each gene by pooling information across genes and shrinking the estimates toward the overall mean of the batch effect estimates across all genes. These parameters are then used to adjust the data for batch effects.[66] Combat was run with default parameters as implemented in the Scanpy[67] function.

MNN or *Mutual Nearest Neighbors* computes a mapping between batches in the feature space. The cells are integrated using the MNN algorithm, which computes pairs of neighboring cells

across pairs of batches (i.e. mutual nearest neighbors) in this space and corrects all cells using kernels combining batch effect estimates between MNN pairs. MNN then iterates through all batches to map onto one reference batch[68]**.** MNN was run using the run mnn function from mnnpy using the default parameters.

Scanorama is an extension of the MNN method. It builds on MNN by simultaneously finding mutual nearest neighbors across all batches using panoramic stitching and embedding these observations into a joint hyperplane[69]. Scanorama was run using the correct_scanpy function with default parameters.

FastMNN is a different implementation of the aforementioned MNN (https://github.com/LTLA/batchelor) algorithm. It performs a multi-sample PCA to reduce dimensionality, identifying MNN pairs in the low-dimensional space, and then correcting the target batch toward the reference using locally weighted correction vectors. (https://marionilab.github.io/FurtherMNN2018/theory/description.html)

Harmony is a method that uses PCA to group the cells into multi-dataset clusters, and then computes cluster-specific linear correction factors. Each cell is then corrected by its cell-specific linear factor using the cluster-weighted average. The method keeps iterating these four steps until cell clusters are stable[70]. Harmony was run using the default parameters.

LIGER or linked inference of genomic experimental relationships uses integrative non-negative matrix factorization (iNMF) to derive and implement a novel coordinate descent algorithm to efficiently do the factorization. This algorithm factorizes each batch matrix into a batch specific and a common matrix. The common matrix is then used as a batch-corrected embedding.[71] LIGER was run using the default parameters of k = 20 and lambda = 5. As LIGER performs its own scaling, we considered it as part of the method and only tested it with unscaled data.

SCALEX [72] is a method for integrating heterogeneous single-cell data online using a variational auto-encoder (VAE) framework. Its generalized encoder disentangles batch-related components from batch-invariant biological components, which are then projected into a common cell-embedding space.

scVI combines a variational autoencoder with a hierarchical Bayesian model. [32] It uses the negative binomial distribution to describe gene expression of each cell, conditioned on unobserved factors and the batch variable. ScVI is run as implemented in Luecken et al.[60]

ScANVI is an extension of scVI that uses an scVI model as pre-training and then performs end-to-end training of a Bayesian semi-supervised cell type classifier on the latent space.[31] By training the model on all cell type labels, scanVI is able to provide an optimized integration. scANVI is run as described in Luecken et al.[60]

Baseline Methods

No Integration serves as a negative control for batch correction metrics, where cells are embedded by PCA on the unintegrated data. A graph is built on this PCA embedding.

No Integration by Batch serves as a negative control, where cells are embedded by computing PCA independently on each batch.

Random Integration serves as a negative control for bio-conservation metrics, but positive control for batch correction metrics. Here, feature values, embedding coordinates, and graph connectivity are all randomly permuted.

Random Integration by celltype serves as a positive control for batch correction metrics and as a negative control for bio-conservation metrics. Here, feature values, embedding coordinates, and graph connectivity are all randomly permuted within each celltype label.

Random Integration by batch serves as a negative control, where feature values, embedding coordinates, and graph connectivity are all randomly permuted within each batch label.

Random Embedding by Celltype serves as a positive control for both batch correction metrics and bio-conservation metrics that rely on cell type labels. Here, cells are embedded as a one-hot encoding of celltype labels.

Random Embedding by Celltype (with jitter) serves as a positive control for both batch correction metrics and bio-conservation metrics that rely on cell type labels, where cells are embedded as a one-hot encoding of celltype labels, with a small amount of random noise added to the embedding.

Random Graph by Celltype serves as a positive control. Here, cells are first embedded as a one-hot encoding of celltype labels. A graph is then built on this embedding.

Metrics

The metrics used for this task cover two different groups of quality of batch integration: the removal of batch effect and biological conservation. The removal of batch effect methods try to measure how well technical variation between batches is removed by the methods while the biological conservation metrics try to capture how well biological effects are conserved after batch removal. Each subtask has separate metrics.

Graph

ARI (Adjusted Rand Index) is a bio-conservation metric and compares the overlap of cell clusters with cell type labels. It considers both correct clustering overlaps while also counting correct disagreements between two clusterings. Using leiden clustering, we test multiple resolutions between 0.1 and 2 for optimal NMI (see below) and use this resolution for ARI computation.

Isolated label F1 is a bio-conservation metric that evaluates how well cell types shared by few batches are separated from others by clustering.The cell type shared by the fewest batches is designated the isolated label For the Isolated label F1 score, the cluster assignment of the isolated label is first optimized (using the F1 score), and the optimal F1 score for the label is then used as the metric. In case of multiple isolated labels, the average score across these labels is used.

NMI (normalized mutual information) is a bio-conservation metric that compares the overlap of two clusterings. Here, the cell-type labels are compared with Louvain clusters computed on the integrated dataset. Louvain clusters are optimized as stated above for the ARI metric.

Graph connectivity is a batch correction metric and assesses whether the kNN graph representation of the integrated data connects all cells with the same cell identity label. The metric measures the proportion of cells in the largest connected component of the graph.

Embedding

Cell Cycle Score is a bio-conservation metric. It is a cell-cycle conservation score that evaluates how similar the variance contributions of cell cycle phase scores are on the embedding before and after integration. Cell cycle phase scores are computed as described in Scanpy tutorials using gene sets from Tirosh et al[73].

Isolated label Silhouette is a bio-conservation metric. It is a score that evaluates the compactness for the labels that are shared by fewest batches (see isolated label F1 above). It indicates how well rare cell types can be preserved after integration.

kBET[74] is a batch correction metric that determines whether the label composition of a k nearest neighborhood (kNN) of a cell is similar to the expected (global) label composition. The test is repeated for a random subset of cells, and the results are summarised as a rejection rate over all tested neighborhoods. Here we run kBet per cell type by subsetting the kNN graph per cell type and using all connected components of this graph. We take the mean over the connected components (that are at least k*3 large), and then take the mean over all cell types.

PC regression (Principal component regression) is a batch correction metric that compares the explained variance by batch before and after integration. It returns a score between 0 and 1 (scaled=True) with 0 if the variance contribution hasn’t changed. The larger the score, the more different the variance contributions are before and after integration.

Batch ASW (Absolute Silhouette Width) is a batch correction metric that computes the silhouette score over all batch labels per cell type. Here, 0 indicates that batches are well mixed and any deviation from 0 indicates there remains a separation between batch labels. This is rescaled to a score between 0 and 1 by taking .

Silhouette (absolute silhouette width) is a bio-conservation metric. It is computed on cell identity labels, measuring their compactness

Feature

HVG conservation (highly variable gene conservation) is a bio-conservation metric. It computes the average percentage of overlapping highly variable genes per batch before and after integration starting from a predefined set of 2000 HVGs that is used for integration.

Task results

Feature subtask

On the feature subtask (**Supplementary Fig. 6)**, Combat on unscaled data performs best, followed by MNN (HVG). Most methods perform best on the pancreas dataset and worst on the lung dataset. Compared to the other subtasks, methods that produce a feature output perform worse compared to the perfect baselines. This seems to be mostly driven by the low scores on the feature specific metric HVG conservation, suggesting that accurate feature reconstruction is a particularly challenging task.

Embedding subtask

The best performing method on the embedding subtask (**Supplementary Fig. 7)** is scANVI in both tested configurations, followed by MNN (hvg/scaled) and Combat (hvg/unscaled). scANVI performs best on the Immune and Lung datasets, while on the pancreas dataset, harmony scores best. Overall, metric scores are higher than in the full feature subtask but lower than in the graph subtask.

As the complexity of the subtasks is in this order, this result is to be expected.

Deep learning methods such as scANVI, scVI or SCALEX rank well on this task, while mutual nearest neighbor approaches such as MNN also seem to score well here.

While Combat and MNN score highly in metrics such as PC regression, cell cycle score and batch ASW where scANVI performs worse, scANVI scores well on ARI where those methods perform worse. MNN and combat seem to perform well on metrics that are computed on the embedding, scANVI seems to perform better on metrics computed on the graph output. This suggests that MNN and ComBat conserve linear biological effects well while removing linear batch effects. However, these methods don’t capture cell type heterogeneity as well as scANVI.

Graph subtask

On the graph subtask scANVI performs best (**Supplementary Fig. 8)**, followed by scVI and Scanorama on scaled data. scANVI is the best performing method on the Immune and the Lung datasets, while Scanorama performs best on the pancreas dataset. It should however be noted that all methods perform well on the pancreas dataset and differences between metric results are small, except for LIGER which performs poorly.

In the graph subtask, all methods perform better than the random baselines across metrics and some methods are quite close in their scores to the perfect baselines, suggesting that the graph subtask is the easiest of the three to solve. Notably, all of the best-performing methods produce

an embedding output which is then used to compute a graph representation, while methods that only produce a graph output (e.g., BBKNN) perform worse. Thus, graph outputs either represent a layer of abstraction of the data in which batch correction is easier to solve, or they are not as rigorously evaluated as embedding outputs by the included metrics. Indeed, some of the metrics such as graph connectivity do not seem to distinguish much between a good and a perfect integration and almost only show the difference between a random output and any integration method.

Across all subtasks scANVI is the top performer, and indeed scANVI does perform well on all batch-corrected outputs that it generates. However, results can not be directly compared to other methods as scANVI considers additional information in the form of cell type labels. Furthermore, across methods and subtasks, using a set of highly variable genes seems to lead to better integration performance than using the full dataset.
Unsurprisingly, the results obtained in this benchmark are quite similar to the results of the recent benchmark on which this task was based[60]. In this task, we judge the methods based only on integration performance, while the Luecken *et al.* study also considers usability and scalability. To extend upon the published benchmark, we added the recently published SCALEX method, which ranks among the top methods for the feature and the embedding subtasks. Further extensions to the batch integration task will include larger dataset sizes to better represent current batch integration efforts that often contain over a million cells[75].

Supplementary Note 1.5: Spatial decomposition

Motivation

Array-based spatial transcriptomics technologies (e.g. ST[76], Visium[77], Dbit-seq[78], Slide-seq[79], HDST[80] and others) measure expression profiles per location in spatial observation units rather than cells. As these units are typically larger than cells, multiple cell types might be contributing to the total measured gene expression in one observation (i.e., spot or bead). Thus, these technologies rely on additional computational analysis to recover the cellular origin of the measured gene expression profile. This is the goal of spatial decomposition or spatial transcriptomics cell-type deconvolution from single cell RNA-seq reference data, as it is also known in the literature.

Task description

The spatial decomposition task revolves around inferring relative cell type abundances in array-based spatial transcriptomics data. Specifically, the task requires methods to estimate the composition of cell identities (i.e., cell type or state) that are present at each capture location (i.e., spot or bead). The cell identity estimates are presented as proportion values, representing

the proportion of the cells at each capture location that belong to a given cell identity. The faithfulness of this inference is evaluated using several metrics. In this task, we distinguish between reference-based decomposition and de novo decomposition, where the former leverages external data (e.g., scRNA-seq or scNuc-seq) to guide the inference process, while the latter only works with the spatial data. In this task, it is required that all datasets have an associated reference single-cell data set to perform reference-based decomposition, but methods are free to ignore this information to perform de novo decomposition instead. All methods benchmarked so far require a scRNA-seq reference to learn the cell-type-specific transcriptomics signature.

Datasets

There are 7 datasets present in the current implementation of the spatial decomposition benchmark:

- 1 dataset consisting of a re-implementation of the ground-truth synthetic data generation from Lopez et al.[81] This dataset contains 5 cell types, 2000 genes and 1600 spots for the spatial data.
- 3 dataset consisting of a synthetic data generation following a dirichlet-based sampling of ground truth mixtures from Andersson et al.[82] from the mouse pancreas dataset[60]. This dataset is the same one as described above. For the spatial ground truth, 100 spots are generated.
- 3 datasets consisting of the same synthetic data generation as described for the mouse pancreas, yet using the tabula muris senis dataset[24]. The subset consists of only the Lung tissues, for mice 30 months old and samples from droplet-based technology. This dataset is described above, and for the spatial ground truth, 100 spots are generated.

While the destVI synthetic dataset accounts for the spatial covariance of cell types, that is, whether cell-types compositions across generated samples follow a specific co-variation (or co-occurrence) pattern. This co-variation is determined by a set of predefined kernels that operates on the synthetic generation step. The other 6 do not and treat each observation as an independent mixture of cell types. In these 6 datasets, cell type heterogeneity is controlled by the concentration parameter alpha from the Dirichlet distribution. The concentration parameter determines the heterogeneity in terms of cell type for the generated spatial samples, which means whether a single sample is composed of more (or fewer) cell-types.

Methods

There are 8 methods so far benchmarked in the task:

Cell2location[83] is a decomposition method based on Negative Binomial regression that is able to account for batch effects in estimating the single-cell gene expression signature used for the spatial decomposition step.

destVI[81] is a decomposition method that leverages a conditional generative model of spatial transcriptomics down to the sub-cell-type variation level, which is then used to decompose the cell-type proportions determining the spatial organization of a tissue.

Stereoscope[82] is a decomposition method based on Negative Binomial regression. It is similar in scope and implementation to cell2location but less flexible to incorporate additional covariates such as batch effects and other types of experimental design annotations.

RCTD[84] (Robust Cell Type Decomposition) is a decomposition method that uses signatures learnt from single-cell data to decompose spatial expression of tissues. It incorporates a platform effect normalization step, which normalizes the scRNA-seq cell type profiles to match the platform effects of the spatial transcriptomics dataset.

NMFreg[85] is a decomposition method combining Non-negative Matrix Factorization (NMF) and Regression, that reconstructs the expression of each spatial location as a weighted combination of cell-type signatures defined by scRNA-seq. It was originally developed for Slide-seq data

NMF is a decomposition method based on Non-negative Matrix Factorization (NMF) that reconstructs the expression of each spatial location as a weighted combination of cell-type signatures defined by scRNA-seq. It is a simpler baseline than NMFreg as it only performs the NMF step based on mean expression signatures of cell types, returning the weights loading of the NMF as (normalized) cell type proportions, without the regression step.

Tangram[86] is a method to map gene expression signatures from scRNA-seq data to spatial data. It performs the cell type decomposition by learning a similarity matrix between single-cell and spatial locations based on gene expression profiles.

NNLS[87] is a decomposition method based on Non-Negative Least Square Regression (NNLS). It was originally introduced by the method AutoGenes[87]

SeuratV3[88] is a decomposition method that is based on Canonical Correlation Analysis (CCA). The joint embedding found by CCA is then used to project labels from the single cell references to the spatial dataset. It was originally introduced by Seurat[88] .

All methods were run based on tutorials of respective implementations. The only methods for which different hyperparameters were used are cell2location and destvi, where various hyperparameters where changed. We recommend checking the github repository for details.

Baseline methods

Random Proportions serves as a positive control, where the predicted proportions are randomly assigned from a Dirichlet distribution.

True Proportions serves as a positive control, where the predicted data is a 1-to-1 copy of the solution data.

Metrics

R2, or the “coefficient of determination”, reports the fraction of the true proportion values’ variance that can be explained by the predicted proportion values. The best score, and upper bound, is 1.0. There is no fixed lower bound for the metric. The uniform/non-weighted average across all cell types/states is used to summarize performance.

Task results

Cell2location is the best performing method across datasets (also robust to hyperparameter settings since variations of cell2location are always top performers; **Supplementary Fig. 9**). The second top performing method is DestVI, which mostly scores second to cell2location. NNLS ,RCTD, and Stereoscope score 3rd, 4th and 5th respectively, In particular, NNLS is the second top performing method in 2 pancreas dataset simulations. In terms of computational resources, cell2location and DestVI are the most time-consuming methods, although this is potentially due to the methods having been designed to run on GPU whereas the benchmark was run on CPU.

Overall, these results suggest that models that can account for covariate effects and are either able to encode prior knowledge or have powerful amortization schemes perform better. While both top performers cell2location and DestVI use the negative binomial distribution as noise model, RCTD and Stereoscope, which also employ suitable models for count data, are not as performant. This suggests that the choice of noise model may be necessary but not sufficient to method performance. Finally, methods such as Seurat V3 and Tangram, which integrate spatial and single-cell data into a low dimensional embedding to perform decomposition, do not seem to be performant nor robust across datasets.
These results broadly recapitulate recently published benchmarks[89,90]. In future extensions of this task, one may include other sources of ground truth, for example by aligning spot-based spatial methods (e.g., 10X Visium) with single-molecule spatial methods (e.g., 10X Xenium) which provide (sub-)cellular resolution.

Supplementary Note 1.6: Denoising

Motivation

scRNA-seq data can be notoriously noisy, with molecular capture rates that often hover around 40% for droplet-based sequencing[9] and up to 95% of measured zeros[10]. In the droplet-based modalities, this low capture rate causes certain reads to be unobserved that may obscure the underlying biological reality of a system or dataset. Since scRNA-seq experiments capture a static snapshot of gene expression within a tissue sample, an additional source of noise in this data type can be attributed to instantaneous variability in expression, creating additional

sparseness in an already lossy modality. To address dropout and other noise, data augmentations that denoise or “impute” scRNA-seq expression matrices have been proposed. Many of these methods use nonlinear or localized smoothing of groups of cells or of genes with similar expression patterns. The approaches for this smoothing and for identification of such groups vary strongly between methods. Although various methodologies and metrics to evaluate these techniques have been introduced for both real and synthetic datasets[91,92], a broader dialogue about whether data denoising should be applied to scRNA-seq data for initial preprocessing, clustering, or downstream analysis is ongoing[93].

Task description

The data denoising task attempts to evaluate the major data denoising tools and to implement reasonable and universal metrics across a variety of datasets. The methods that are considered take as input a scRNA-seq expression matrix, which is then randomly partitioned into “train” and “test” subsets using the molecular cross validation (MCV) approach[11]. MCV creates train and test splits by simulating two random samples from the observed reads in each cell of the dataset. Once the training set has been denoised, its similarity to the testing set is assessed via one of several loss functions. Although datasets are assumed to already contain only the cells and genes that pass initial pre-processing steps, further normalization is considered a part of the evaluated method. To facilitate the comparison of model performance using MCV, each denoising method is applied to the “train” subset, and model outputs are evaluated against the “test” subset using various metrics.

Datasets

The methods in the denoising task are currently applied only to droplet-based sequencing experiments and within expression matrices. As there are few requirements on datasets to which denoising methods can be applied, we have selected datasets from diverse tissues. The three datasets currently included in this dataset include:

10x 1k Peripheral Blood Mononuclear Cells (PBMCs) consists of 1087 PBMCs from a healthy donor, including measurements of 15099 genes. These data is supplied without cell type labels. These data were sequenced on 10X v3 chemistry in November 2018 by 10X Genomics.

Pancreas (inDrop) includes 1,937 Human pancreatic islet cells × 15575 genes, including 14 labelled cell types. These data were collected in a single batch via the inDrop sequencing protocol.

Tabula Muris Senis Lung includes all lung cells from Tabula Muris Senis, a 500k cell-atlas from 18 organs and tissues across the mouse lifespan. The lung cells are all collected via 10X sequencing, yielding 24,540 cells × 17,985 genes across 3 time points and 39 cell types.

Methods

Currently benchmarked methods include ALRA, MAGIC, DCA, KNN-smoothing, and iterative KNN smoothing.

ALRA[94] (Adaptively-thresholded Low Rank Approximation) is a method for imputation of missing values in single cell RNA-sequencing data. Given a normalized scRNA-seq expression matrix, it first imputes values using rank-*k* approximation, via singular value decomposition. Next, a symmetric distribution is fitted to the near-zero imputed values for each gene (row) of the matrix. The right "tail" of this distribution is then used to threshold the accepted nonzero entries. This same threshold is then used to rescale the matrix, once the "biological zeros" have been removed. This yields an imputed output matrix. Upon addition of this method to the Open Problems denoising task, it was noted that ALRA's performance is highly sensitive to the order of data normalizations: performance of the MSE and Poisson loss metrics differs greatly if log/sqrt transformation is placed before (the standard/regular order) or after library size normalization (reverse order). Likewise, although the authors of ALRA suggest using a log transform when preprocessing the data, earlier versions of Open Problems found that a square-root transform may perform better. Four versions of ALRA are included in the current iteration of the OpenProblems benchmark, including regular and reverse order normalization steps, and including square root and log transformations.

MAGIC[95] (Markov Affinity-based Graph Imputation of Cells) is a method for imputation and denoising of noisy or dropout-prone single cell RNA-sequencing data. Given a normalized scRNA-seq expression matrix, it first calculates Euclidean distances between each pair of cells in the dataset, which is then augmented using a Gaussian kernel (function) and row-normalized to give a normalized affinity matrix. A t-step markov process is then calculated, by powering this affinity matrix *t* times. Finally, the powered affinity matrix is right-multiplied by the normalized data, causing the final imputed values to take the value of a per-gene average weighted by the affinities of cells. The resultant imputed matrix is then rescaled, to more closely match the magnitude of measurements in the normalized (input) matrix. A faster version of MAGIC, which performs an imputation on the principle components of the scRNA-seq expression matrix and then projects the approximated imputation results back into the original dimensionality/scale of the input dataset, is also implemented in the current distribution of MAGIC. Both the standard version and approximate versions are included in the current version of the OpenProblems benchmark. Similarly to ALRA, MAGIC is also sensitive to the order of the square root and library size normalization steps. Likewise, four total versions of MAGIC are included in the current iteration of the OpenProblems benchmark, including regular and reverse order normalization steps, and including regular and approximate (PC-based) imputation.

DCA [96] (Deep Count Autoencoder) is a method to remove the effect of dropout in scRNA-seq data. DCA takes into account the count structure, overdispersed nature and sparsity of scRNA-seq data types using a deep autoencoder with a zero-inflated negative binomial (ZINB) loss. The autoencoder is then applied to the dataset, where the mean of the fitted negative binomial distributions is used to fill each entry of the imputed matrix.

KNN-smoothing is a method for denoising data based on the k-nearest neighbors. Given a normalized scRNA-seq matrix, KNN-smoothing calculates a *k*-nearest neighbor matrix using Euclidean distances between cell pairs. Each cell's denoised expression is then defined as the average expression of each of its neighbors.

Iterative kNN-smoothing [97] is a method to repair or denoise noisy scRNA-seq expression matrices. Given a scRNA-seq expression matrix, KNN-smoothing first applies initial normalization and smoothing. Then, a chosen number of principal components is used to calculate Euclidean distances between cells. Minimally sized neighborhoods are initially determined from these Euclidean distances, and expression profiles are shared between neighboring cells. Then, the resultant smoothed matrix is used as input to the next step of smoothing, where the size *(k)* of the considered neighborhoods is increased, leading to greater smoothing. This process continues until a chosen maximum *k* value has been reached, at which point the iteratively smoothed object is then optionally scaled to yield a final result.

Baseline Methods

No Denoising serves as a negative control, where the denoised data is a copy of the unaltered training data. This represents the scoring threshold if denoising was not performed on the data.

Perfect denoising serves as a positive control, where the test data is copied 1-to-1 to the denoised data. This makes it seem as if the data is perfectly denoised as it will be compared to the test data in the metrics.

Metrics

Two metrics are used: mean squared error (MSE) and the Poisson or "count-based" loss.

MSE measures the mean squared error between the denoised counts of the training dataset and the true counts of the test dataset after reweighting by the train/test ratio.

Poisson measures the Poisson log likelihood of observing the true counts of the test dataset given the distribution given in the denoised dataset.

Task results

The reversed normalization versions of MAGIC are the best performing methods across datasets, yielding nearly perfect scores (0.98) from the Poisson loss metric, as well as the highest observed overall MSE loss values (0.28; **Supplementary Fig. 10**). The standard normalization-order versions of MAGIC also achieve similar MSE values, but show poorer performance under Poisson loss (~0.55). The next best performing method overall is the square root and reversed normalization order version of ALRA, which performs significantly less well under MSE (-0.01) but matches the very accurate Poisson loss of the best performing MAGIC versions (0.98). The standard normalization-order versions of MAGIC have the next highest

overall performance scores, with mean scores of 0.49 and 0.41.Notably, ALRA suffers from the greatest memory requirements and runtime of the included models, potentially due to a lack of multithreading support.

Across datasets, the non-standard square root transform and reverse normalization order implementations of methods performed best for the denoising task. This ranking predominantly depended on the Poisson metric. Specifically, we note that the square root transform is a variance stabilising transform for the Poisson distribution. Moreover, further analysis showed that reversing normalization order pushes values < 1 closer to zero, which suggests that the Poisson loss is highly affected by low, non-zero values.

The results of this task can currently only be interpreted in the context of droplet-based data. Extensions to more datasets, and especially to those including different assay chemistries/sequencing protocols as well as more tissues and greater numbers of experimental batches/greater experimental segmentation will be necessary to create a more generalizable benchmark. Further extensions to multimodal data and even plate-based data may be considered, in order to extend the utility of this benchmark. Multiple additional methods, many of which have been published recently[98–100] would further enrich the current task. Especially as the Poisson metric appears to strongly affect the evaluation of preprocessing decisions, additional metrics that evaluate different distributional assumptions (i.e. negative binomial) should also be considered.

Finally, the task would benefit from extension to additional subtasks, such as benchmarking approaches that consider the preservation of class/cluster separation, correlations of denoised values with bulk RNA-sequencing values, and robustness of denoising tools to batch effects within real or simulated data. Existing benchmarks of denoising and imputation tools use several additional metrics not encapsulated by the current Open Problems benchmark, and encompass additional tools not currently included in this task.

Supplementary Note 1.7: Matching modalities

Motivation

Cellular function is regulated by the complex interplay of different types of biological molecules (DNA, RNA, proteins, etc.), which determine the state of a cell. Several recently described technologies[101,102] allow for simultaneous measurement of different aspects of cellular state (e.g., DNA accessibility and RNA expression, or RNA and protein levels). These technologies enable us to better understand cellular function, however datasets are still rare and there are tradeoffs that these measurements make for profiling multiple modalities. Joint methods can be more expensive, lower throughput, or more noisy than measuring a single modality at a time.

Therefore it is useful to develop methods that are capable of matching measurements of the same biological system but obtained using different technologies on different cells.

Task description

In this task, the goal is to learn a latent space where cells profiled by different technologies in different modalities are matched if they have the same state. We use jointly profiled data as ground truth so that we can evaluate when the observations from the same cell acquired using different modalities are similar. A perfect result has each of the paired observations sharing the same coordinates in the latent space. A method that can achieve this would be able to match datasets across modalities to enable multimodal cellular analysis from separately measured profiles.

Datasets

In its current state, this task has three datasets from two different multimodal technologies:

CITE-seq is a multimodal extension for the 10x scRNA-seq platform that measures cellular transcriptomes and surface proteins in the same cells. We used a CITE-seq dataset of 8,000 cord blood mononuclear cells, which was profiled using a panel of 13 protein-targeting antibodies[102].

sciCAR is a combinatorial indexing-based assay that jointly measures cellular transcriptomes and the accessibility of cellular chromatin in the same cells. Here, we use two sciCAR datasets that were obtained from the same study[101]. The first dataset contains 4,825 cells from three cell lines (HEK293T cells, NIH/3T3 cells, and A549 cells) at multiple timepoints (0, 1 hour, 3 hours) after dexamethasone treatment. The second dataset contains 11,233 cells from wild-type adult mouse kidney.

Methods

Harmonic alignment[103] embeds cellular data from each modality into a common space by computing a mapping between the 100-dimensional diffusion maps[43] of each modality. This mapping is computed by computing an isometric transformation of the eigenmaps, and concatenating the resulting diffusion maps together into a joint 200-dimensional space. This joint diffusion map space is used as output for the task.

Mutual nearest neighbors (MNN) embeds cellular data from each modality into a common space by computing a mapping between modality-specific 100-dimensional SVD embeddings. The embeddings are integrated using the FastMNN version (https://github.com/LTLA/batchelor) of the MNN algorithm[68], which generates an embedding of the second modality mapped to the SVD space of the first. This corrected joint SVD space is used as output for the task.

Procrustes superimposition[104] embeds cellular data from each modality into a common space by aligning the 100-dimensional SVD embeddings to one another by using an isomorphic transformation that minimizes the root mean squared distance between points. The unmodified SVD embedding and the transformed second modality are used as output for the task.

Baseline Methods

Random Features serves as a negative control, where 20-dimensional SVD is computed on the first modality, and is then randomly permuted twice, once for use as the output for each modality.

True Features serves as a positive control, where 20-dimensional SVD is computed on the first modality, and this same embedding is used as output for both modalities.

Metrics

Two complimentary metrics were used to assess the proximity of cell embeddings from different modalities: k-nearest-neighbor (kNN) area under the curve (AUC) and the mean squared error (MSE).

kNN-AUC: The kNN-AUC measures the percentage overlap of local cellular neighborhoods of modality 1 and modality 2. The AUC is calculated over the number of neighbors in these neighborhoods. A higher score indicates a better matching.

MSE: Mean squared error (MSE) is the average distance between each pair of matched observations of the same cell in the learned latent space. Lower is better.

Task results

Procrustes superimposition is the top-performing matching modality method across datasets (**Supplementary Fig. 11**). It is the top performer on the CITE-seq and sciCAR mouse kidney datasets, and it has only a slightly worse MSE than MNN on the sciCAR cell lines. Furthermore, it is comparatively fast on the tested datasets.

However, the highest mean score that Procrustes superimposition achieves is 0.37 on the CITE-seq data, suggesting that procrustes is still far from optimal matching performance, which is indicated by a mean metric score of 1 (a mean score of 0 indicating random performance). Indeed, on the sciCAR mouse kidney dataset procrustes only scores 0.05, which is slightly better than random performance. This suggests that none of the tested methods perform well on this task.

Finally, the results of this stub task suggest that methods perform far better on the CITE-seq dataset than on multiome datasets. However, as only 3 datasets are currently added to this task (1 CITE, 2 multiome datasets), further datasets must be added to validate this observation.

1.8 Perturbation prediction

Motivation

Single-cell studies of disease typically aim to identify cell states that have a causal role in disease pathogenesis. Targeting such cell states via chemical compounds should combat the underlying disease. Thus, understanding the causal links between chemical perturbations and cell state changes is crucial for drug development. However, traditional perturbation experiments are costly and time-consuming, limiting the exploration of potential therapeutic targets. By accurately predicting the effects of chemical perturbations, single-cell data science can accelerate and expand the development of new medicines.

Task description

In this task, methods are challenged to predict how chemical compounds alter gene expression in a given cell type. The methods take as input the statistical significance of a change in gene expression calculated via differential gene expression analysis on a pseudobulked scRNA-seq expression matrix before and after the chemical perturbation. The data is split into a train and test set so that each cell type and each compound is present in the training set, but some cell type and compound combinations are held out and must be predicted. Models predict gene expression change for each gene, specifically $-\ log_{10}(pval)sign(lFC)$, where $pval$ indicates the p-value and lFC the log-fold-change. A perfect prediction matches the values calculated on the observed test set data. A method that performs well could be used to inform perturbation experiment design and improve in silico screening of compounds in drug discovery.

Datasets

NeurIPS 2023 PBMC is a novel, custom designed and generated single-cell perturbational dataset (SRA: SRP527159, GEO: GSE279945) in human peripheral blood mononuclear cells (PBMCs). The authors selected 146 compounds from the Library of Integrated Network-Based Cellular Signatures (LINCS) Connectivity Map dataset[105] and measured single-cell gene expression profiles after 24 hours of treatment. The experiment was repeated in three healthy human donors, and the compounds were selected based on diverse and robust transcriptional signatures observed in CD34+ hematopoietic stem cells (data not released). The dataset comprises annotated T-cells, B-cells, myeloid cells, and NK cells. To supplement this dataset, joint scRNA and single-cell chromatin accessibility measurements were taken at baseline for each donor using the 10x Multiome assay.

Methods

Methods included in this benchmark were developed for the NeurIPS 2023 perturbation prediction competition that was run on Kaggle. As such, they are only described in the respective competition paper[106] and not in separate publications.

Metrics

Mean Row-wise Root Mean Squared Error was computed between the predicted and measured p-values. A perfect prediction would yield 0 MRRMSE. There is no upper bound to the error.

Task results

Simpler methods, such as a feedforward neural network retrained with pseudo-labels and gradient-boosted decision trees, tended to outperform more complex methods, such as RNN/CNN ensembles and transformer ensembles. However, even the predictions from the best performing methods were still far from ground truth. As this task was submitted in an existing manuscript exactly as implemented in Open Problems, we refer to the manuscript for further description and analysis of the results[106].

1.9 Spatially variable genes

Motivation

Recent years have witnessed significant progress in spatially-resolved transcriptome profiling techniques that simultaneously characterize cellular gene expression and their physical position, generating spatial transcriptomic (ST) data. The application of these techniques has dramatically advanced our understanding of disease and developmental biology by uncovering the spatial patterning of these processes. Uncovering such spatial patterns is the raison d'etre of ST technologies and as such, many computational workflows have been, and are being, developed to identify these signals. The first step of any workflow to uncover spatially relevant patterns of gene expression is to select genes that exhibit spatial patterns. These genes, defined as spatially variable genes (SVGs), contain additional information about the spatial structure of the tissues of interest, compared to highly variable genes (HVGs) that are typically selected to analyse cellular variation in dissociated scRNA-seq data.

Task description

Identification of spatially variable genes is crucial for studying spatial domains within tissue microenvironments, developmental gradients and cell signalling pathways. In this task we attempt to evaluate various methods for detecting SVGs using a number of realistic simulated datasets with diverse patterns derived from real-world spatial transcriptomics data using scDesign3. Methods are provided with simulated data consisting of realistic gene expression data and spatial coordinates, and must output a spatial variability score per gene that is correlated with true spatial variability score input into the scDesign3 simulation.

Datasets

We collected a total of 50 real-world datasets across 17 different tissue types and nine technologies, including 10X Visium (n = 20), Slide-seqV2 (n = 5), Slide-tag (n = 4), DBiT-seq (n = 6), Stereo-seq (n = 5), 10X Xenium (n = 2), MERFISH (n = 5), seqFISH (n = 1), and STARmap (n = 2). The mitochondrial and lowly expressed genes for each dataset were filtered. For each dataset, a Gaussian Process (GP) model is built using scDesign3 framework. Synthetic data generated by mixing the GP model and a non-spatial model (obtained by shuffling mean parameters of the GP model to remove spatial correlation between spots) to generate gene expressions with various spatial variability. Details about the spatial datasets and simulation approaches are available in the preprint for the benchmark[107].

Methods

14 methods for detecting spatially variable genes were tested. These overlap exactly with the methods described in the relevant benchmarking manuscript[107].

Baseline Methods

Random score serves as a negative control, where the spatial variability score for each gene was randomly assigned.

Metrics

Kendall rank correlation was used to assess the similarity of the predicted spatial variability score with the true score input into the simulation that balances the GP and the non-spatial model simulation. The correlation has a maximum value of 1 if the two ranks perfectly agree with each other, and otherwise, a minimum value of -1.

Task results

Detailed results of this task have been submitted for review and are described in the corresponding manuscript[107].

References

1. Cannoodt, R. *et al.* Viash: A meta-framework for building reusable workflow modules. *J. Open Source Softw.* **9**, 6089 (2024).
2. Di Tommaso, P. *et al.* Nextflow enables reproducible computational workflows. *Nat. Biotechnol.* **35**, 316–319 (2017).
3. Attardi, J. *Using Gatsby and Netlify CMS*. (Apress).
4. Virshup, I., Rybakov, S., Theis, F. J., Angerer, P. & Wolf, F. A. anndata: Access and store annotated data matrices. *Journal of Open Source Software* **9**, 4371 (2024).
5. Maier-Hein, L. *et al.* Why rankings of biomedical image analysis competitions should be interpreted with care. *Nat. Commun.* **9**, 5217 (2018).
6. Cannoodt*, R. *et al.* funkyheatmap: Visualising data frames with mixed data types. *J. Open Source Softw.* **10**, 7698 (2025).
7. Dimitrov, D. *et al.* Comparison of methods and resources for cell-cell communication inference from single-cell RNA-Seq data. *Nat. Commun.* **13**, 3224 (2022).
8. Luecken, M. D. *et al.* Benchmarking atlas-level data integration in single-cell genomics. *Nat. Methods* **19**, 41–50 (2022).
9. Hagemann-Jensen, M. *et al.* Single-cell RNA counting at allele and isoform resolution using Smart-seq3. *Nat. Biotechnol.* **38**, 708–714 (2020).
10. Hicks, S. C., Townes, F. W., Teng, M. & Irizarry, R. A. Missing data and technical variability in single-cell RNA-sequencing experiments. *Biostatistics* **19**, 562–578 (2018).
11. Batson, J., Royer, L. & Webber, J. Molecular Cross-Validation for Single-Cell RNA-seq. *bioRxiv* 786269 (2019) doi:10.1101/786269.
12. Gigante, S. *et al. Openproblems-Bio/openproblems: v1.0.0*. (Zenodo, 2024). doi:10.5281/ZENODO.13769879.
13. Almet, A. A., Cang, Z., Jin, S. & Nie, Q. The landscape of cell-cell communication through

single-cell transcriptomics. *Curr Opin Syst Biol* **26**, 12–23 (2021).

14. Elosua-Bayes, M., Nieto, P., Mereu, E., Gut, I. & Heyn, H. SPOTlight: seeded NMF regression to deconvolute spatial transcriptomics spots with single-cell transcriptomes. *Nucleic Acids Res.* **49**, e50 (2021).
15. Tasic, B. *et al.* Adult mouse cortical cell taxonomy revealed by single cell transcriptomics. *Nat. Neurosci.* **19**, 335–346 (2016).
16. Wu, S. Z. *et al.* A single-cell and spatially resolved atlas of human breast cancers. *Nat. Genet.* **53**, 1334–1347 (2021).
17. Badia-I-Mompel, P. *et al.* decoupleR: ensemble of computational methods to infer biological activities from omics data. *Bioinform Adv* **2**, vbac016 (2022).
18. Efremova, M., Vento-Tormo, M., Teichmann, S. A. & Vento-Tormo, R. CellPhoneDB: inferring cell-cell communication from combined expression of multi-subunit ligand-receptor complexes. *Nat. Protoc.* **15**, 1484–1506 (2020).
19. Wang, Y. *et al.* iTALK: an R Package to Characterize and Illustrate Intercellular Communication. *bioRxiv* 507871 (2019) doi:10.1101/507871.
20. Hou, R., Denisenko, E., Ong, H. T., Ramilowski, J. A. & Forrest, A. R. R. Predicting cell-to-cell communication networks using NATMI. *Nat. Commun.* **11**, 5011 (2020).
21. Raredon, M. S. B. *et al.* Computation and visualization of cell-cell signaling topologies in single-cell systems data using Connectome. *Sci. Rep.* **12**, 4187 (2022).
22. Cabello-Aguilar, S. *et al.* SingleCellSignalR: inference of intercellular networks from single-cell transcriptomics. *Nucleic Acids Res.* **48**, e55 (2020).
23. Kolde, R., Laur, S., Adler, P. & Vilo, J. Robust rank aggregation for gene list integration and meta-analysis. *Bioinformatics* **28**, 573–580 (2012).
24. Tabula Muris Consortium. A single-cell transcriptomic atlas characterizes ageing tissues in the mouse. *Nature* **583**, 590–595 (2020).
25. Taylor, S. R. *et al.* Molecular topography of an entire nervous system. *Cell* **184**,

4329–4347.e23 (2021).

26. Wagner, D. E. *et al.* Single-cell mapping of gene expression landscapes and lineage in the zebrafish embryo. *Science* **360**, 981–987 (2018).
27. Lun, A. T. L., Bach, K. & Marioni, J. C. Pooling across cells to normalize single-cell RNA sequencing data with many zero counts. *Genome Biol.* **17**, 75 (2016).
28. Cover, T. & Hart, P. Nearest neighbor pattern classification. *IEEE Trans. Inf. Theory* **13**, 21–27 (1967).
29. Hosmer, D. W. & Lemeshow, S. Applied Logistic Regression. Preprint at https://doi.org/10.1002/0471722146 (2000).
30. Hinton, G. E. Connectionist learning procedures. *Artif. Intell.* **40**, 185–234 (1989).
31. Xu, C. *et al.* Probabilistic harmonization and annotation of single-cell transcriptomics data with deep generative models. *Mol. Syst. Biol.* **17**, e9620 (2021).
32. Lopez, R., Regier, J., Cole, M. B., Jordan, M. I. & Yosef, N. Deep generative modeling for single-cell transcriptomics. *Nature Methods* vol. 15 1053–1058 Preprint at https://doi.org/10.1038/s41592-018-0229-2 (2018).
33. Lotfollahi, M. *et al.* Query to reference single-cell integration with transfer learning. *bioRxiv* (2020) doi:10.1101/2020.07.16.205997.
34. Hao, Y. *et al.* Integrated analysis of multimodal single-cell data. *Cell* **184**, 3573–3587.e29 (2021).
35. Chen, T. & Guestrin, C. XGBoost. in *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining* (ACM, New York, NY, USA, 2016). doi:10.1145/2939672.2939785.
36. Abdelaal, T. *et al.* A comparison of automatic cell identification methods for single-cell RNA sequencing data. *Genome Biol.* **20**, 194 (2019).
37. Pliner, H. A., Shendure, J. & Trapnell, C. Supervised classification enables rapid annotation of cell atlases. *Nat. Methods* **16**, 983–986 (2019).

38. Chari, T. & Pachter, L. The Specious Art of Single-Cell Genomics. *bioRxiv* 2021.08.25.457696 (2022) doi:10.1101/2021.08.25.457696.
39. Olsson, A. *et al.* Single-cell analysis of mixed-lineage states leading to a binary cell fate choice. *Nature* **537**, 698–702 (2016).
40. Nestorowa, S. *et al.* A single-cell resolution map of mouse hematopoietic stem and progenitor cell differentiation. *Blood* **128**, e20–31 (2016).
41. Xiong, J., Gong, F., Wan, L. & Ma, L. NeuralEE: A GPU-Accelerated Elastic Embedding Dimensionality Reduction Method for Visualizing Large-Scale scRNA-Seq Data. *Front. Genet.* **11**, 786 (2020).
42. Pearson, K. LIII. On lines and planes of closest fit to systems of points in space. *The London, Edinburgh, and Dublin Philosophical Magazine and Journal of Science* **2**, 559–572 (1901).
43. Coifman, R. R. & Lafon, S. Diffusion maps. *Appl. Comput. Harmon. Anal.* **21**, 5–30 (2006).
44. Moon, K. R. *et al.* Visualizing structure and transitions in high-dimensional biological data. *Nat. Biotechnol.* **37**, 1482–1492 (2019).
45. Agrawal, A., Ali, A. & Boyd, S. Minimum-Distortion Embedding. *arXiv [cs.LG]* (2021).
46. van der Maaten, L. & Hinton, G. Visualizing Data using t-SNE. *J. Mach. Learn. Res.* **9**, 2579–2605 (2008).
47. McInnes, L., Healy, J. & Melville, J. UMAP: Uniform Manifold Approximation and Projection for Dimension Reduction. *arXiv [stat.ML]* (2018).
48. Narayan, A., Berger, B. & Cho, H. Assessing single-cell transcriptomic variability through density-preserving data visualization. *Nat. Biotechnol.* **39**, 765–774 (2021).
49. Schober, P., Boer, C. & Schwarte, L. A. Correlation Coefficients: Appropriate Use and Interpretation. *Anesth. Analg.* **126**, 1763–1768 (2018).
50. Venna, J. & Kaski, S. Neighborhood preservation in nonlinear projection methods: An experimental study. in *Artificial Neural Networks — ICANN 2001* 485–491 (Springer Berlin

Heidelberg, Berlin, Heidelberg, 2001).

51. Zhang, Y., Shang, Q. & Zhang, G. pyDRMetrics - A Python toolkit for dimensionality reduction quality assessment. *Heliyon* **7**, e06199 (2021).
52. Chen, L. & Buja, A. Local Multidimensional Scaling for Nonlinear Dimension Reduction, Graph Drawing, and Proximity Analysis. *J. Am. Stat. Assoc.* **104**, 209–219 (2009).
53. Vrahatis, A. G., Tasoulis, S. K., Dimitrakopoulos, G. N. & Plagianakos, V. P. Visualizing High-Dimensional Single-Cell RNA-seq Data via Random Projections and Geodesic Distances. in *2019 IEEE Conference on Computational Intelligence in Bioinformatics and Computational Biology (CIBCB)* 1–6 (2019).
54. Klimovskaia, A., Lopez-Paz, D., Bottou, L. & Nickel, M. Poincaré maps for analyzing complex hierarchies in single-cell data. *Nat. Commun.* **11**, 2966 (2020).
55. Tarashansky, A. J., Xue, Y., Li, P., Quake, S. R. & Wang, B. Self-assembling manifolds in single-cell RNA sequencing data. *Elife* **8**, (2019).
56. Ding, J. & Regev, A. Deep generative model embedding of single-cell RNA-Seq profiles on hyperspheres and hyperbolic spaces. *Nat. Commun.* **12**, 2554 (2021).
57. Quinn, T. P. & Erb, I. Amalgams: data-driven amalgamation for the dimensionality reduction of compositional data. *NAR Genom Bioinform* **2**, lqaa076 (2020).
58. Cooley, S. M., Hamilton, T., Aragones, S. D., Ray, J. C. J. & Deeds, E. J. A novel metric reveals previously unrecognized distortion in dimensionality reduction of scRNA-seq data. *bioRxiv* (2019) doi:10.1101/689851.
59. Zappia, L., Phipson, B. & Oshlack, A. Exploring the single-cell RNA-seq analysis landscape with the scRNA-tools database. *PLoS Comput. Biol.* **14**, e1006245 (2018).
60. Luecken, M. D. *et al.* Benchmarking atlas-level data integration in single-cell genomics. *Cold Spring Harbor Laboratory* 2020.05.22.111161 (2020) doi:10.1101/2020.05.22.111161.
61. Tran, H. T. N. *et al.* A benchmark of batch-effect correction methods for single-cell RNA sequencing data. *Genome Biol.* **21**, 12 (2020).

62. Chazarra-Gil, R., van Dongen, S., Kiselev, V. Y. & Hemberg, M. Flexible comparison of batch correction methods for single-cell RNA-seq using BatchBench. *Nucleic Acids Res.* **49**, e42 (2021).
63. Mereu, E. *et al.* Benchmarking single-cell RNA-sequencing protocols for cell atlas projects. *Nat. Biotechnol.* **38**, 747–755 (2020).
64. Vieira Braga, F. A. *et al.* A cellular census of human lungs identifies novel cell states in health and in asthma. *Nat. Med.* **25**, 1153–1163 (2019).
65. Polański, K. *et al.* BBKNN: fast batch alignment of single cell transcriptomes. *Bioinformatics* **36**, 964–965 (2020).
66. Johnson, W. E., Li, C. & Rabinovic, A. Adjusting batch effects in microarray expression data using empirical Bayes methods. *Biostatistics* **8**, 118–127 (2007).
67. Wolf, F. A., Angerer, P. & Theis, F. J. SCANPY: large-scale single-cell gene expression data analysis. *Genome Biol.* **19**, 15 (2018).
68. Haghverdi, L., Lun, A. T. L., Morgan, M. D. & Marioni, J. C. Batch effects in single-cell RNA-sequencing data are corrected by matching mutual nearest neighbors. *Nat. Biotechnol.* **36**, 421–427 (2018).
69. Hie, B., Bryson, B. & Berger, B. Efficient integration of heterogeneous single-cell transcriptomes using Scanorama. *Nat. Biotechnol.* **37**, 685–691 (2019).
70. Korsunsky, I. *et al.* Fast, sensitive and accurate integration of single-cell data with Harmony. *Nat. Methods* **16**, 1289–1296 (2019).
71. Welch, J. D. *et al.* Single-Cell Multi-omic Integration Compares and Contrasts Features of Brain Cell Identity. *Cell* **177**, 1873–1887.e17 (2019).
72. Xiong, L. *et al.* Online single-cell data integration through projecting heterogeneous datasets into a common cell-embedding space. *Nat. Commun.* **13**, 6118 (2022).
73. Tirosh, I. *et al.* Dissecting the multicellular ecosystem of metastatic melanoma by single-cell RNA-seq. *Science* **352**, 189–196 (2016).

74. Büttner, M., Miao, Z., Wolf, F. A., Teichmann, S. A. & Theis, F. J. A test metric for assessing single-cell RNA-seq batch correction. *Nat. Methods* **16**, 43–49 (2019).
75. Sikkema, L. *et al.* An integrated cell atlas of the human lung in health and disease. *Nat. Med. (accepted)* (2023) doi:10.1101/2022.03.10.483747.
76. Ståhl, P. L. *et al.* Visualization and analysis of gene expression in tissue sections by spatial transcriptomics. *Science* **353**, 78–82 (2016).
77. Salmén, F. *et al.* Barcoded solid-phase RNA capture for Spatial Transcriptomics profiling in mammalian tissue sections. *Nat. Protoc.* **13**, 2501–2534 (2018).
78. Liu, Y. *et al.* High-Spatial-Resolution Multi-Omics Sequencing via Deterministic Barcoding in Tissue. *Cell* **183**, 1665–1681.e18 (2020).
79. Stickels, R. R. *et al.* Highly sensitive spatial transcriptomics at near-cellular resolution with Slide-seqV2. *Nat. Biotechnol.* (2020) doi:10.1038/s41587-020-0739-1.
80. Vickovic, S. *et al.* High-definition spatial transcriptomics for in situ tissue profiling. *Nat. Methods* **16**, 987–990 (2019).
81. Lopez, R. *et al.* DestVI identifies continuums of cell types in spatial transcriptomics data. *Nat. Biotechnol.* (2022) doi:10.1038/s41587-022-01272-8.
82. Andersson, A. *et al.* Single-cell and spatial transcriptomics enables probabilistic inference of cell type topography. *Commun Biol* **3**, 565 (2020).
83. Kleshchevnikov, V. *et al.* Comprehensive mapping of tissue cell architecture via integrated single cell and spatial transcriptomics. *Cold Spring Harbor Laboratory* 2020.11.15.378125 (2020) doi:10.1101/2020.11.15.378125.
84. Cable, D. M. *et al.* Robust decomposition of cell type mixtures in spatial transcriptomics. *Nat. Biotechnol.* (2021) doi:10.1038/s41587-021-00830-w.
85. Rodriques, S. G. *et al.* Slide-seq: A scalable technology for measuring genome-wide expression at high spatial resolution. *Science* **363**, 1463–1467 (2019).
86. Biancalani, T. *et al.* Deep learning and alignment of spatially resolved single-cell

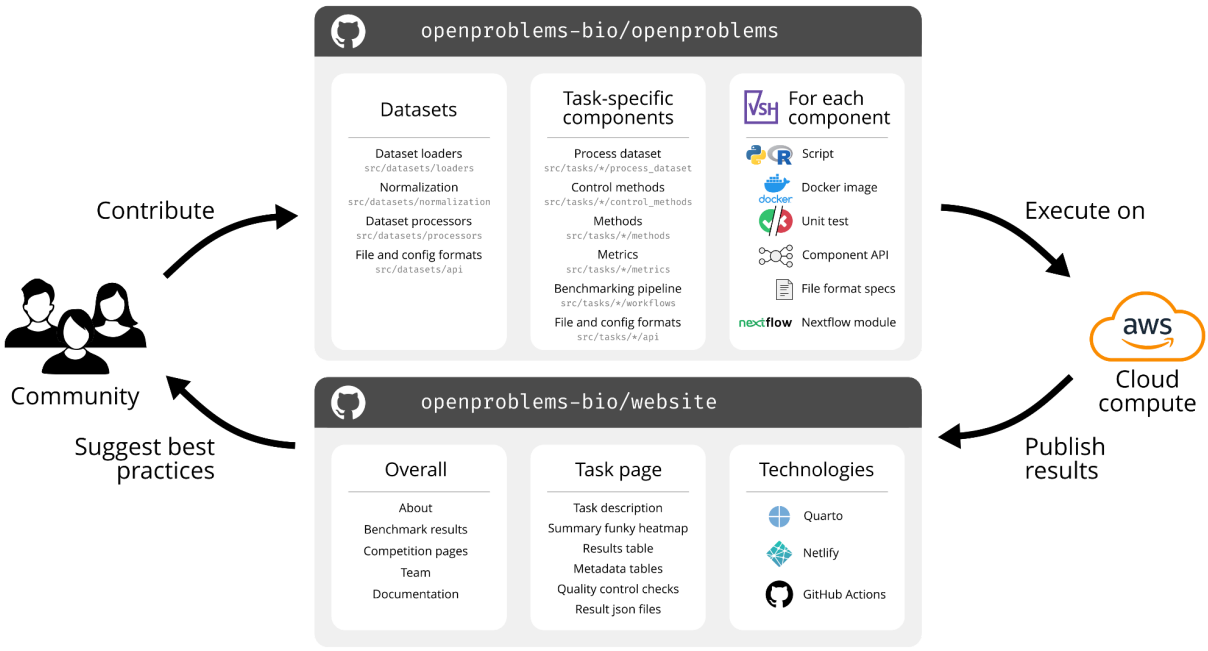
transcriptomes with Tangram. *Nat. Methods* **18**, 1352–1362 (2021).

87. Aliee, H. & Theis, F. J. AutoGeneS: Automatic gene selection using multi-objective optimization for RNA-seq deconvolution. *Cell Syst* **12**, 706–715.e4 (2021).
88. Butler, A., Hoffman, P., Smibert, P., Papalexi, E. & Satija, R. Integrating single-cell transcriptomic data across different conditions, technologies, and species. *Nat. Biotechnol.* **36**, 411–420 (2018).
89. Li, B. *et al.* Benchmarking spatial and single-cell transcriptomics integration methods for transcript distribution prediction and cell type deconvolution. *Nat. Methods* 1–9 (2022).
90. Sang-aram, C., Browaeys, R., Seurinck, R. & Saeys, Y. Spotless: a reproducible pipeline for benchmarking cell type deconvolution in spatial transcriptomics. *bioRxiv* 2023.03.22.533802 (2023) doi:10.1101/2023.03.22.533802.
91. Dai, C. *et al.* scIMC: a platform for benchmarking comparison and visualization analysis of scRNA-seq data imputation methods. *Nucleic Acids Res.* **50**, 4877–4899 (2022).
92. Hou, W., Ji, Z., Ji, H. & Hicks, S. C. A systematic evaluation of single-cell RNA-sequencing imputation methods. *Genome Biol.* **21**, 218 (2020).
93. Luecken, M. D. & Theis, F. J. Current best practices in single-cell RNA-seq analysis: a tutorial. *Mol. Syst. Biol.* **15**, e8746 (2019).
94. Linderman, G. C., Zhao, J. & Kluger, Y. Zero-preserving imputation of scRNA-seq data using low-rank approximation. *bioRxiv* (2018) doi:10.1101/397588.
95. van Dijk, D. *et al.* Recovering Gene Interactions from Single-Cell Data Using Data Diffusion. *Cell* **174**, 716–729.e27 (2018).
96. Eraslan, G., Simon, L. M., Mircea, M., Mueller, N. S. & Theis, F. J. Single-cell RNA-seq denoising using a deep count autoencoder. *Nat. Commun.* **10**, 1–14 (2019).
97. Wagner, F., Yan, Y. & Yanai, I. K-nearest neighbor smoothing for high-throughput single-cell RNA-Seq data. *bioRxiv* (2017) doi:10.1101/217737.
98. Zhu, X. *et al.* Dimensionality reduction of Single-cell RNA sequencing data by combining

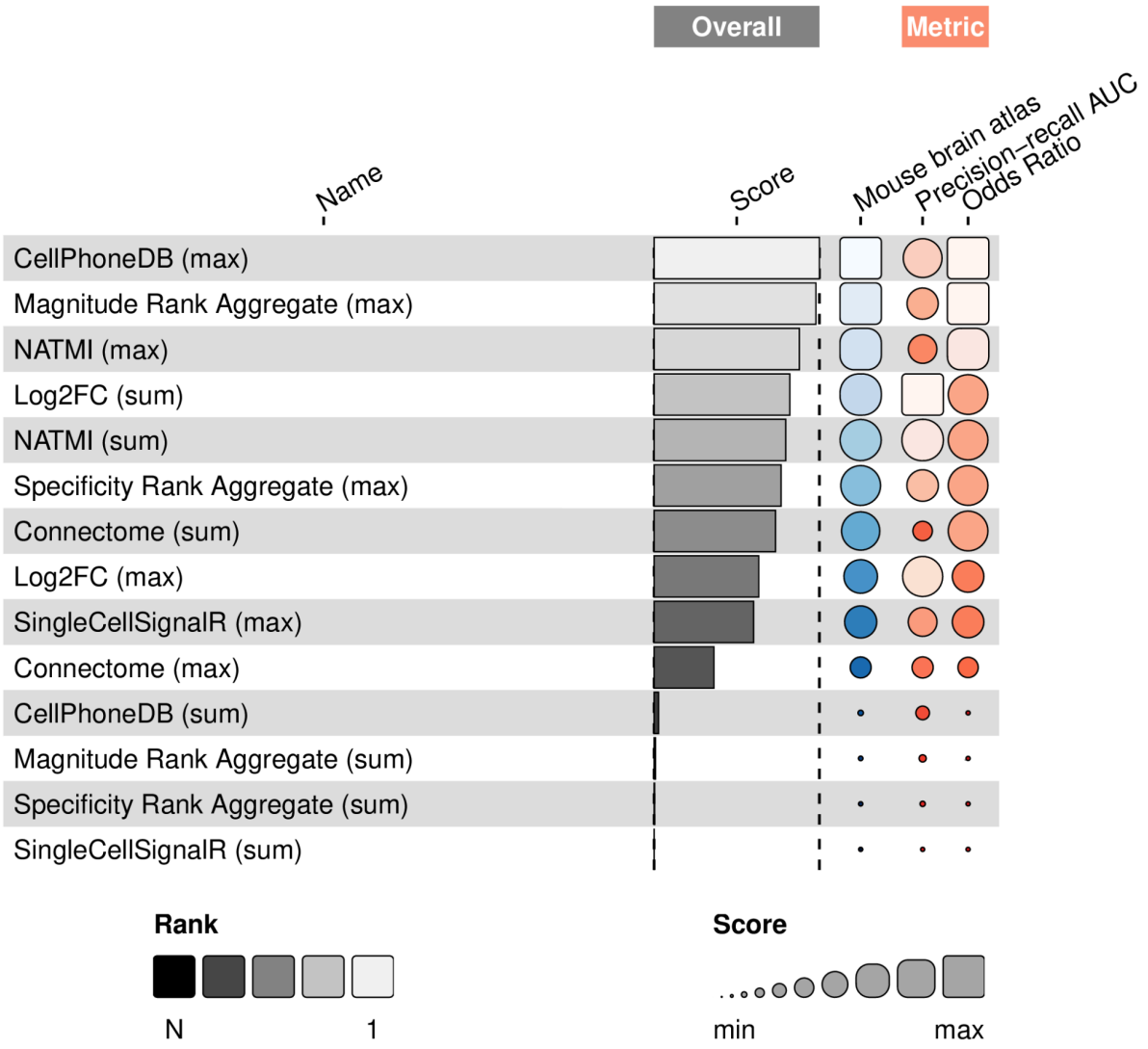
entropy and denoising AutoEncoder. *J. Comput. Biol.* **29**, 1074–1084 (2022).

99. Wang, X. *et al.* Local nonlinear dimensionality reduction via preserving the geometric structure of data. *Pattern Recognit.* **143**, 109663 (2023).
100. Su, E. C.-Y. & Wu, H.-M. Dimension reduction and visualization of multiple time series data: a symbolic data analysis approach. *Comput. Stat.* **39**, 1937–1969 (2024).
101. Cao, J. *et al.* Joint profiling of chromatin accessibility and gene expression in thousands of single cells. *Science* **361**, 1380–1385 (2018).
102. Stoeckius, M. *et al.* Simultaneous epitope and transcriptome measurement in single cells. *Nat. Methods* **14**, 865–868 (2017).
103. Stanley, J. S., III, Gigante, S., Wolf, G. & Krishnaswamy, S. Harmonic Alignment. *arXiv [cs.LG]* (2018).
104. Gower, J. C. Generalized procrustes analysis. *Psychometrika* **40**, 33–51 (1975).
105. Subramanian, A. *et al.* A next generation Connectivity Map: L1000 platform and the first 1,000,000 profiles. *Cell* **171**, 1437–1452.e17 (2017).
106. Szałata, A. *et al.* A benchmark for prediction of transcriptomic responses to chemical perturbations across cell types. Preprint at https://openreview.net/forum?id=WTI4RJYSVm&noteId=ywZk9zB7Vc.
107. Li, Z. *et al.* Benchmarking computational methods to identify spatially variable genes and peaks. *bioRxivorg* (2023) doi:10.1101/2023.12.02.569717.

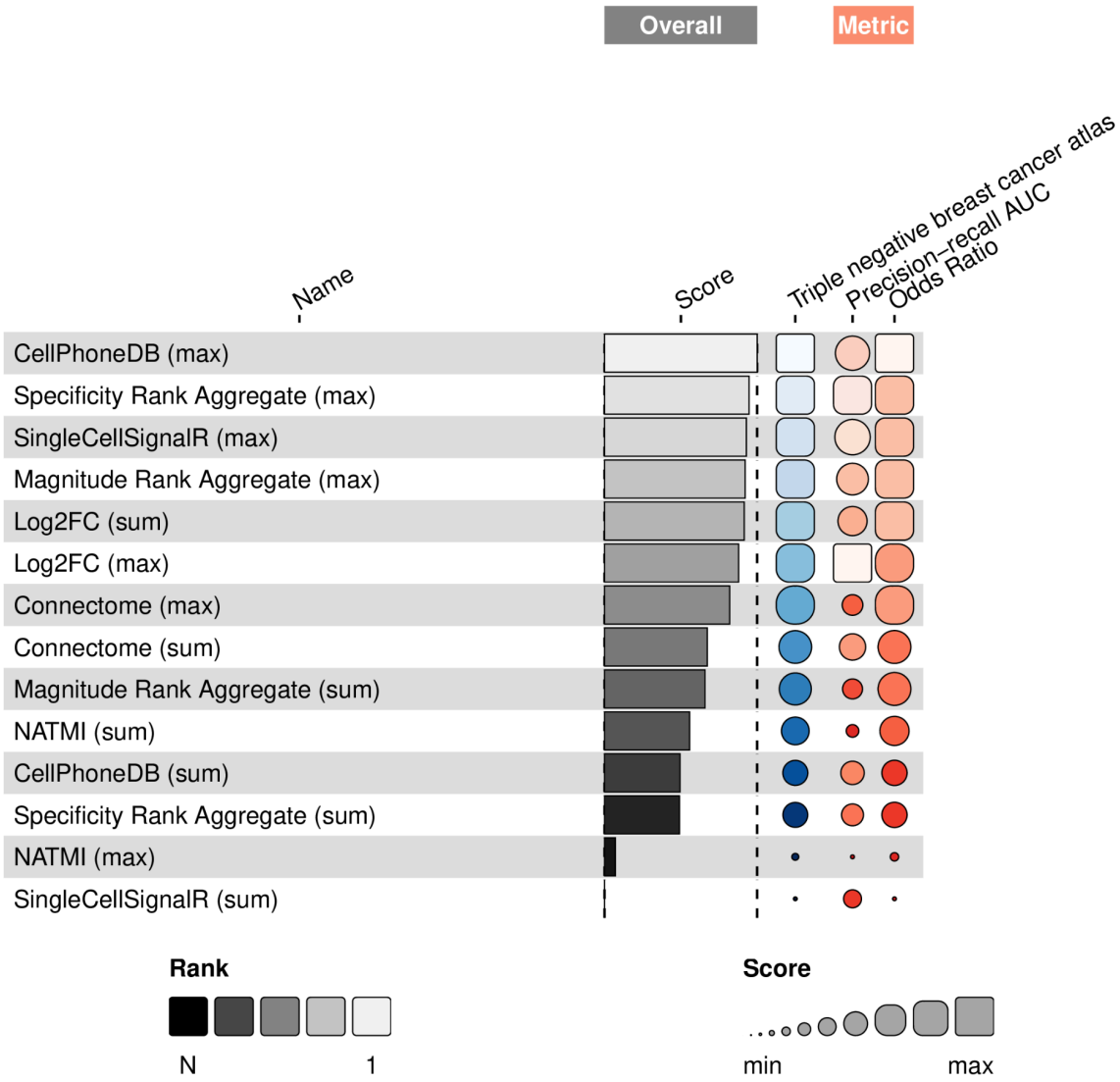
Supplementary Figures



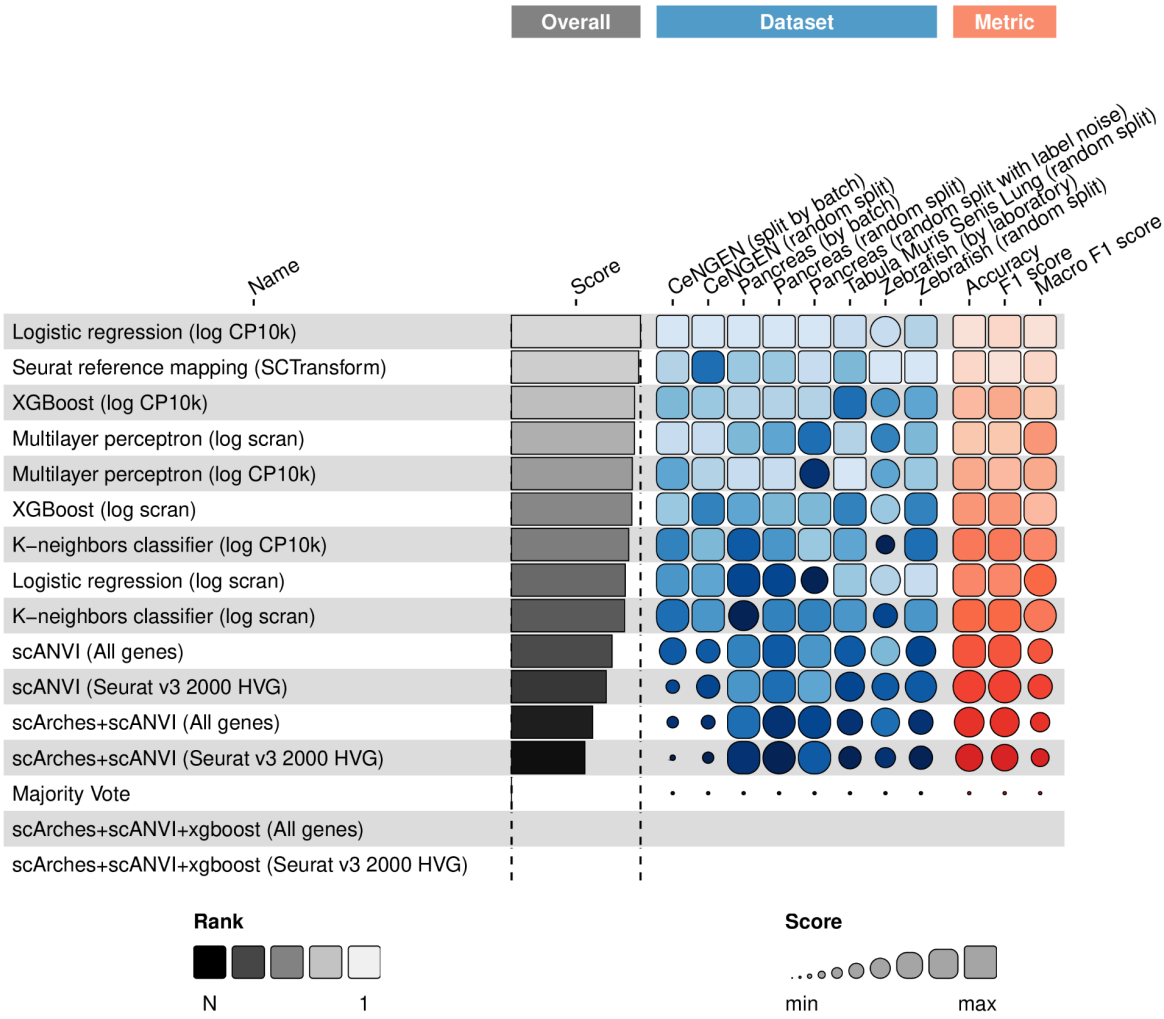
Supplementary Figure 1: Setup of the Open Problems Infrastructure. The Open problems github repo receives community contributions as pull requests. GitHub Actions continuous integration workflows tests all of the contributed Viash components and builds them into standalone executables with versioned Docker images. Dataset processing and benchmark workflows are run on AWS Batch. Finished benchmarking runs are contributed to the website repo as JSON files, where they are parsed and integrated into the website that is rendered using Quarto. The website is hosted via Netlify, and contains method rankings that suggest best practices to the community.



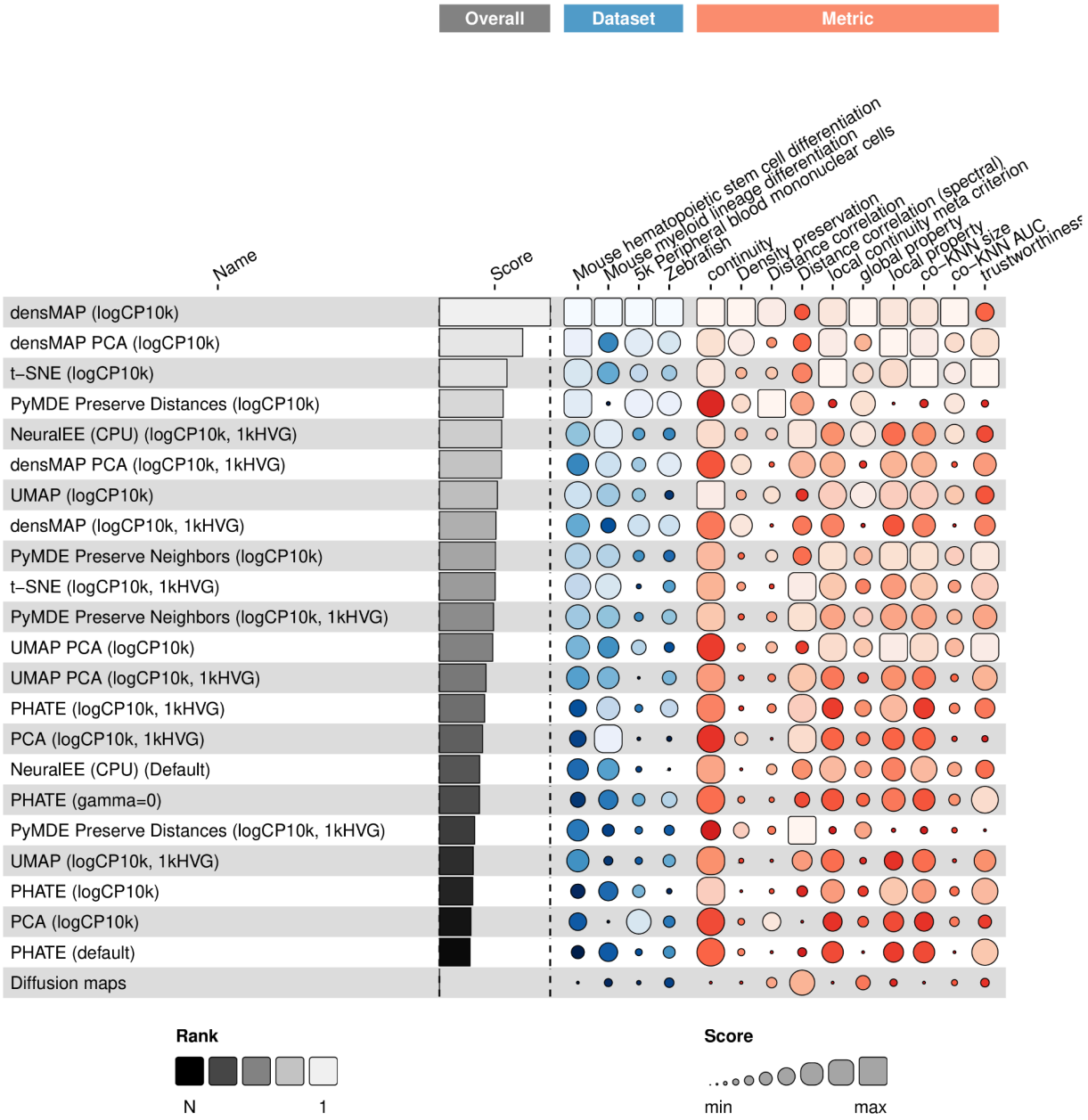
Supplementary Figure 2: Results of the CCC source-target task. Methods are ranked using a mean over all dataset scores. Dataset scores in turn are computed as the baseline-normalized mean of all metrics (**Methods**). The size of circles show baseline-normalized metric (or metric aggregation) scores, and the color of the circles indicates the method rank from a particular metric.



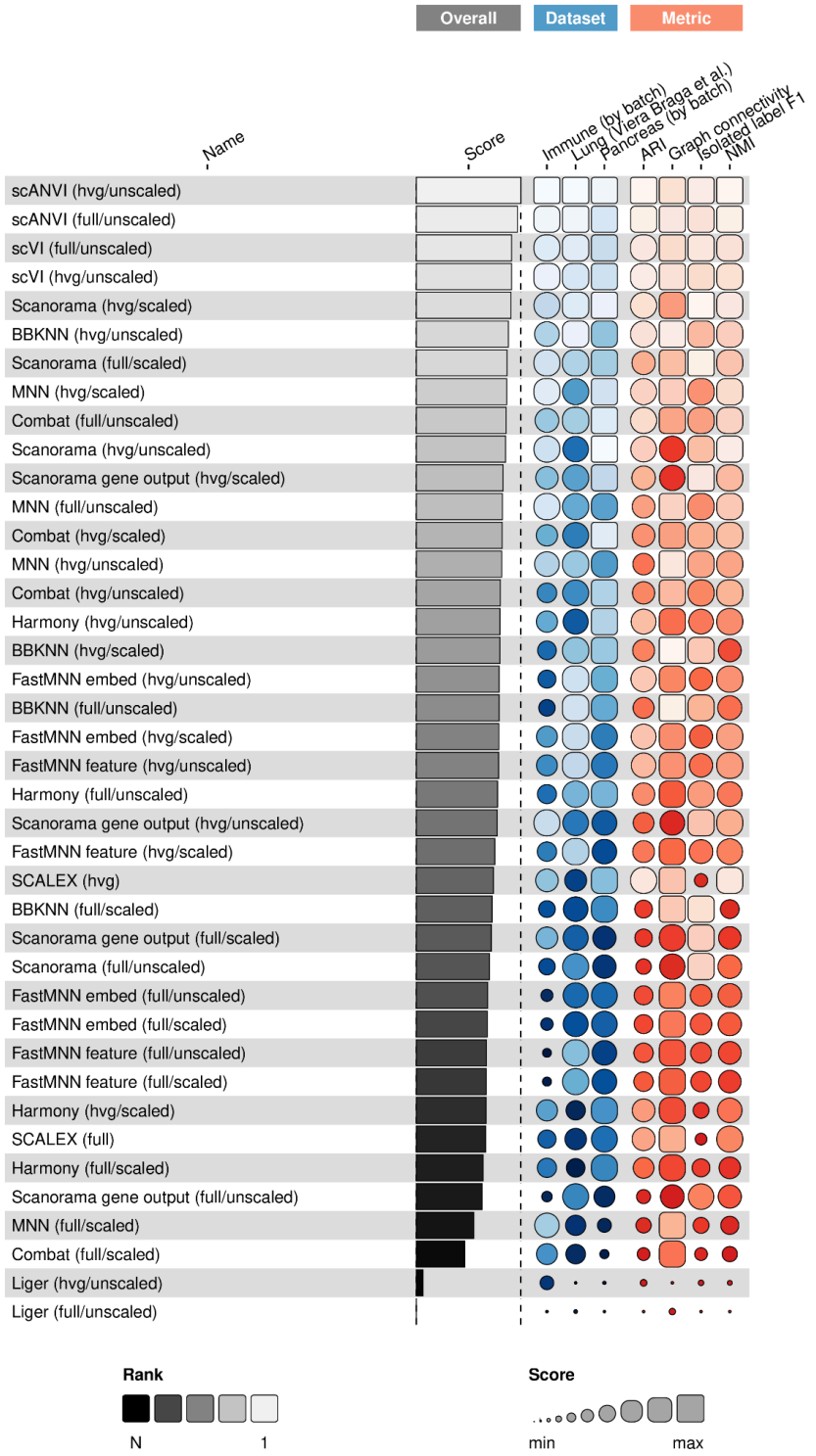
Supplementary Figure 3: Results of the CCC ligand-target task. Methods are ranked using a mean over all dataset scores. Dataset scores in turn are computed as the baseline-normalized mean of all metrics (**Methods**). The size of circles show baseline-normalized metric (or metric aggregation) scores, and the color of the circles indicates the method rank from a particular metric.



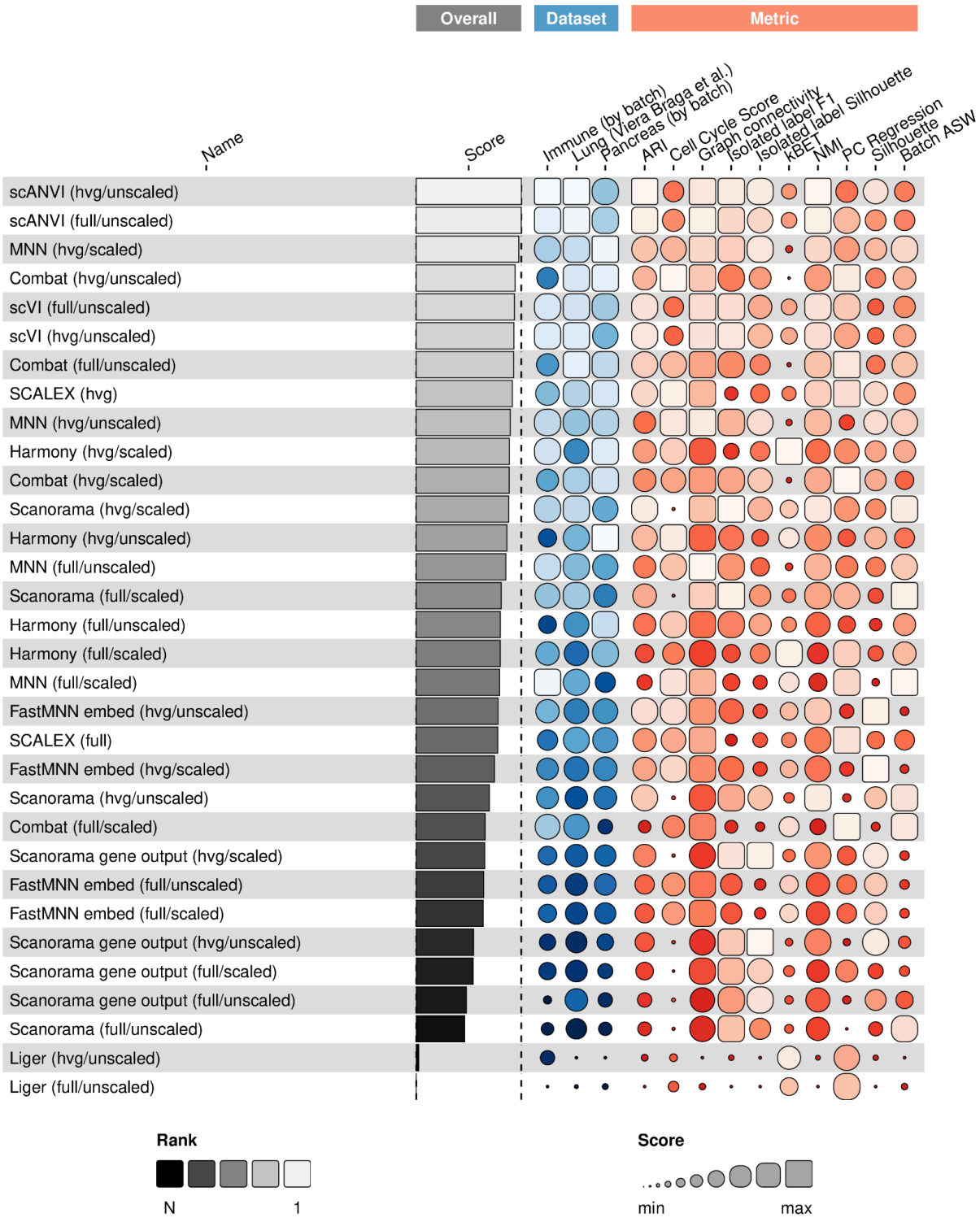
Supplementary Figure 4: Results of the label projection task. Methods are ranked using a mean over all dataset scores. Dataset scores in turn are computed as the baseline-normalized mean of all metrics (**Methods**). The size of circles show baseline-normalized metric (or metric aggregation) scores, and the color of the circles indicates the method rank from a particular metric.



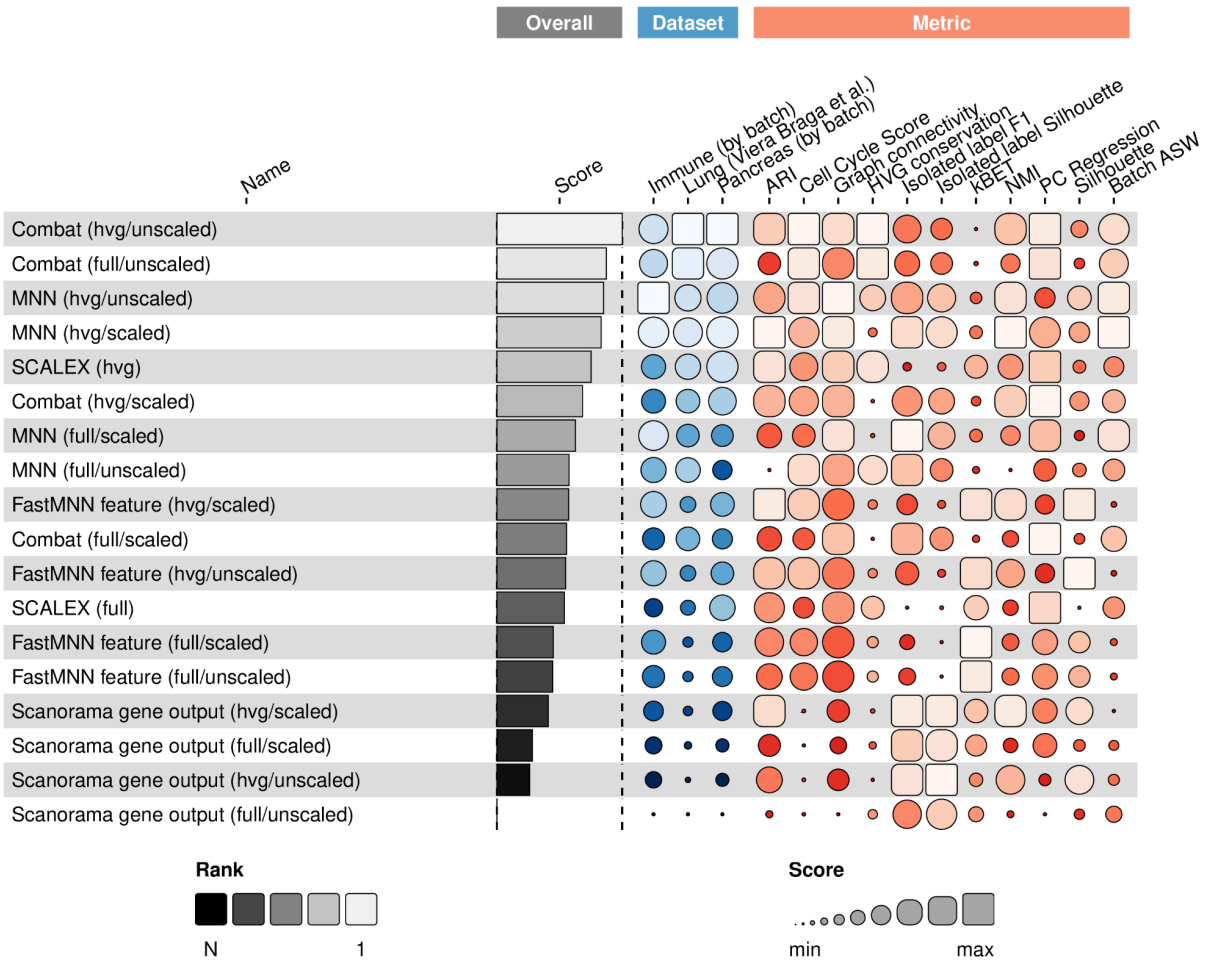
Supplementary Figure 5: Results of the dimensionality reduction for 2D visualization task. Methods are ranked using a mean over all dataset scores. Dataset scores in turn are computed as the baseline-normalized mean of all metrics (**Methods**). The size of circles show baseline-normalized metric (or metric aggregation) scores, and the color of the circles indicates the method rank from a particular metric.



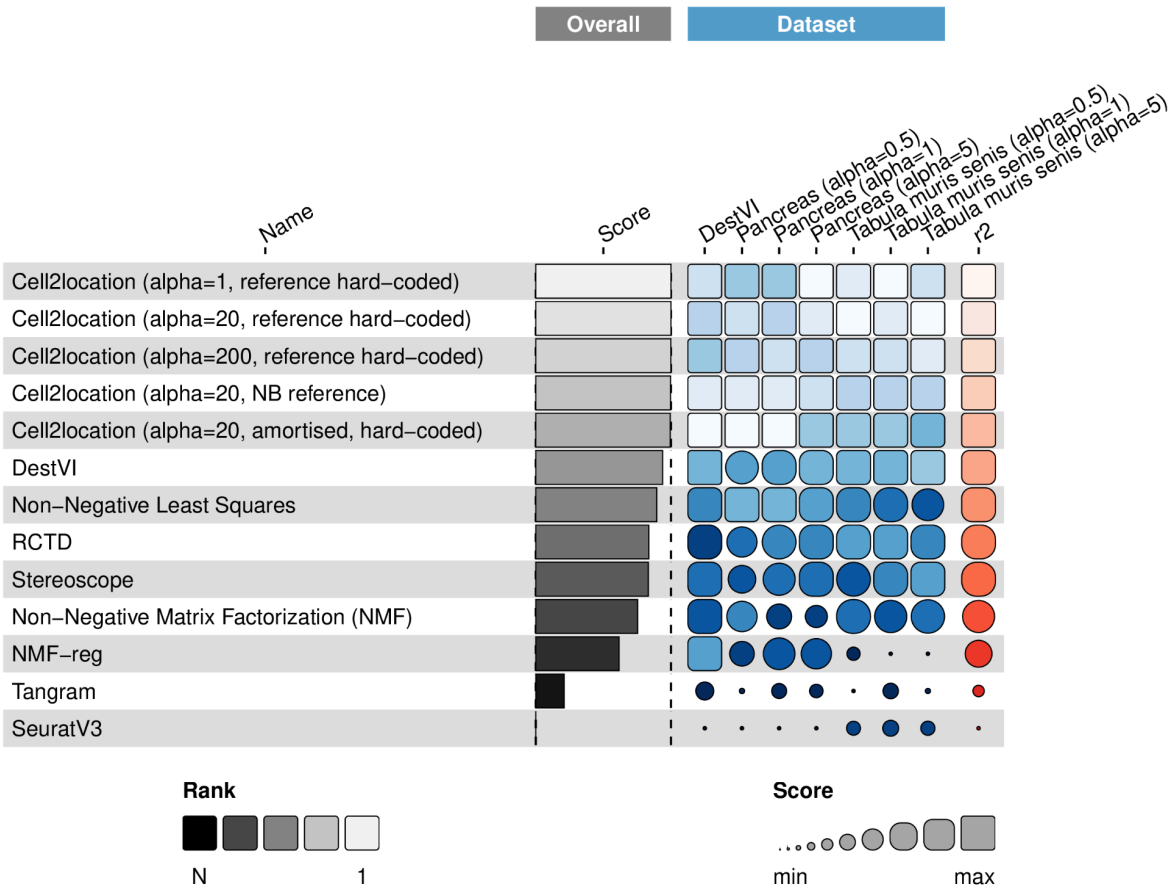
Supplementary Figure 6: Results of the batch integration graph subtask. Methods are ranked using a mean over all dataset scores. Dataset scores in turn are computed as the baseline-normalized mean of all metrics (**Methods**). The size of circles show baseline-normalized metric (or metric aggregation) scores, and the color of the circles indicates the method rank from a particular metric.



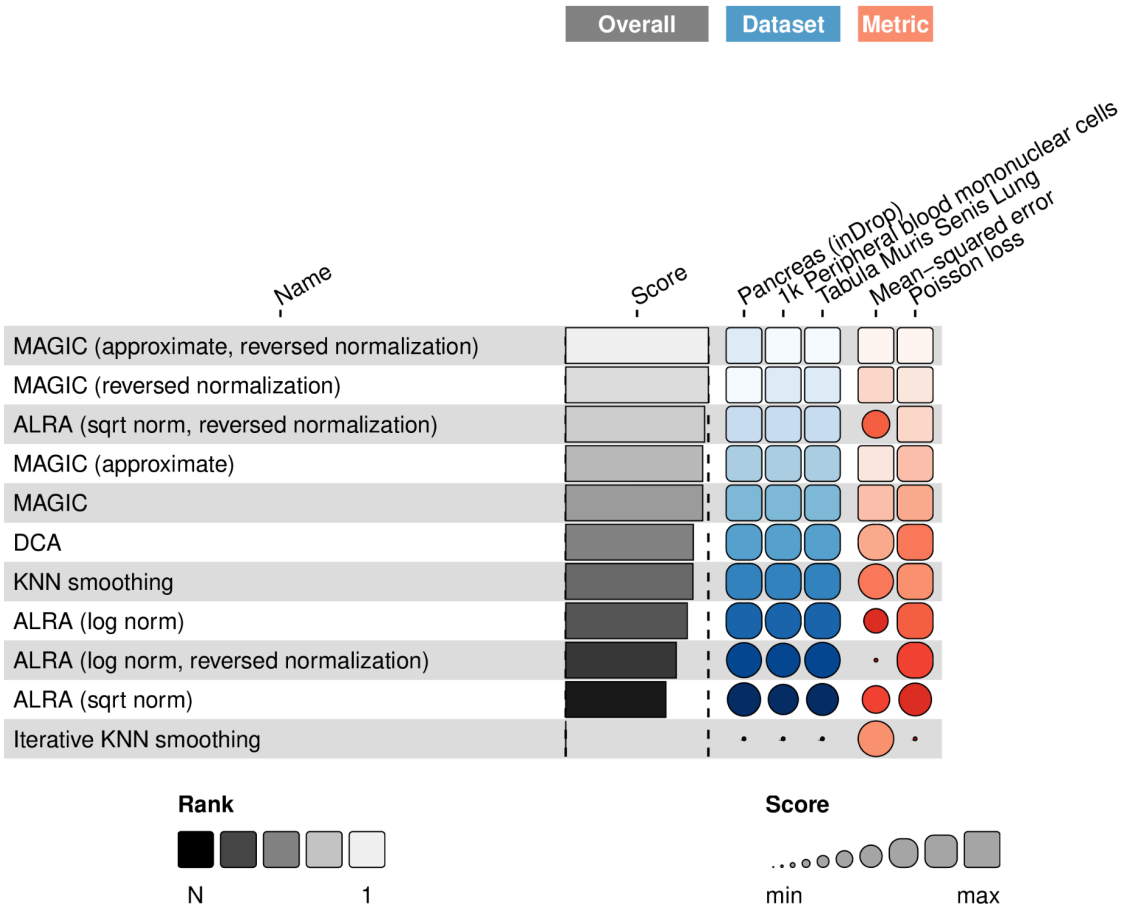
Supplementary Figure 7: Results of the batch integration embedding subtask. Methods are ranked using a mean over all dataset scores. Dataset scores in turn are computed as the baseline-normalized mean of all metrics (**Methods**). The size of circles show baseline-normalized metric (or metric aggregation) scores, and the color of the circles indicates the method rank from a particular metric.



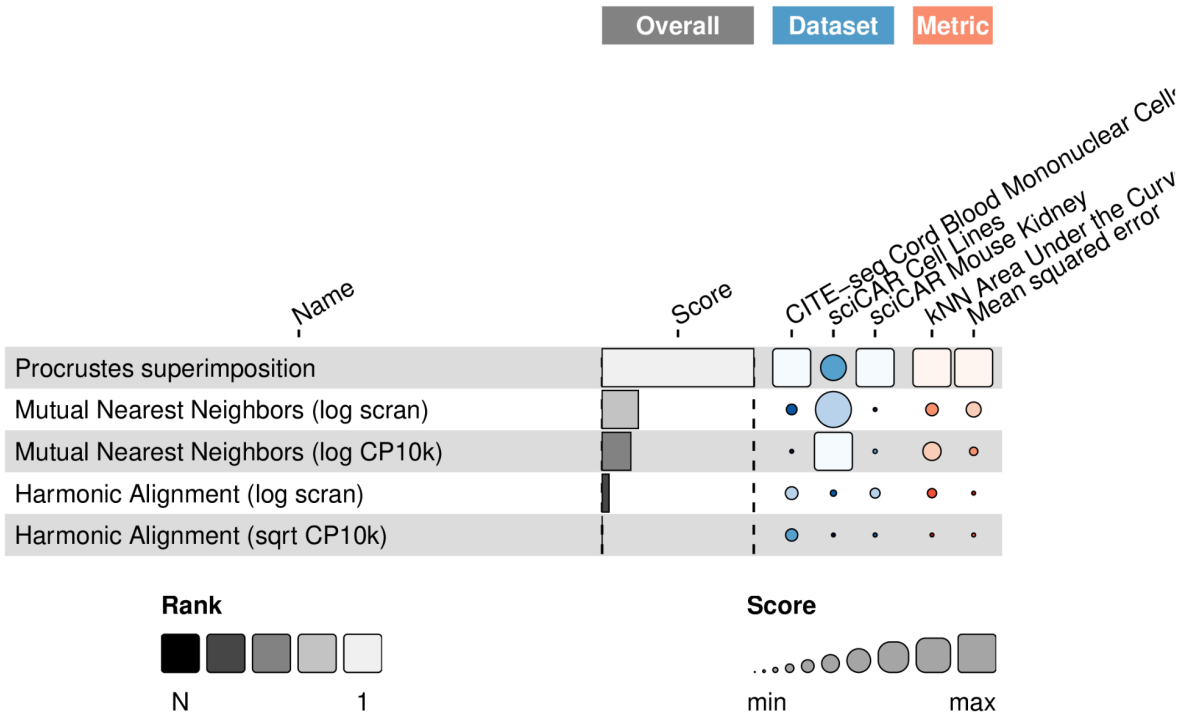
Supplementary Figure 8: Results of the batch integration feature subtask. Methods are ranked using a mean over all dataset scores. Dataset scores in turn are computed as the baseline-normalized mean of all metrics (**Methods**). The size of circles show baseline-normalized metric (or metric aggregation) scores, and the color of the circles indicates the method rank from a particular metric.



Supplementary Figure 9: Results of the spatial decomposition task. Methods are ranked using a mean over all dataset scores. Dataset scores in turn are computed as the baseline-normalized mean of all metrics (**Methods**). The size of circles show baseline-normalized metric (or metric aggregation) scores, and the color of the circles indicates the method rank from a particular metric.



Supplementary Figure 10: Results of the denoising task. Methods are ranked using a mean over all dataset scores. Dataset scores in turn are computed as the baseline-normalized mean of all metrics (**Methods**). The size of circles show baseline-normalized metric (or metric aggregation) scores, and the color of the circles indicates the method rank from a particular metric.



Supplementary Figure 11: Results of the matching modalities task. Methods are ranked using a mean over all dataset scores. Dataset scores in turn are computed as the baseline-normalized mean of all metrics (**Methods**). The size of circles show baseline-normalized metric (or metric aggregation) scores, and the color of the circles indicates the method rank from a particular metric.