

A Technical Appendices and Supplementary Material

A.1 scGeneScope Data Generation

Here, we give a detailed overview of the experimental data generation process leading to the scGeneScope dataset.

A.1.1 Technical Note on the Treatment of Replicates

In the main text of our paper we employ a simplified usage of “replicate”, which we define as “two samples from the same biological population, which are treated in the same way yet may be measured with separate measurement techniques”. While we believe that this simplification will help a generalist audience follow our work more readily, we acknowledge that it glosses over key

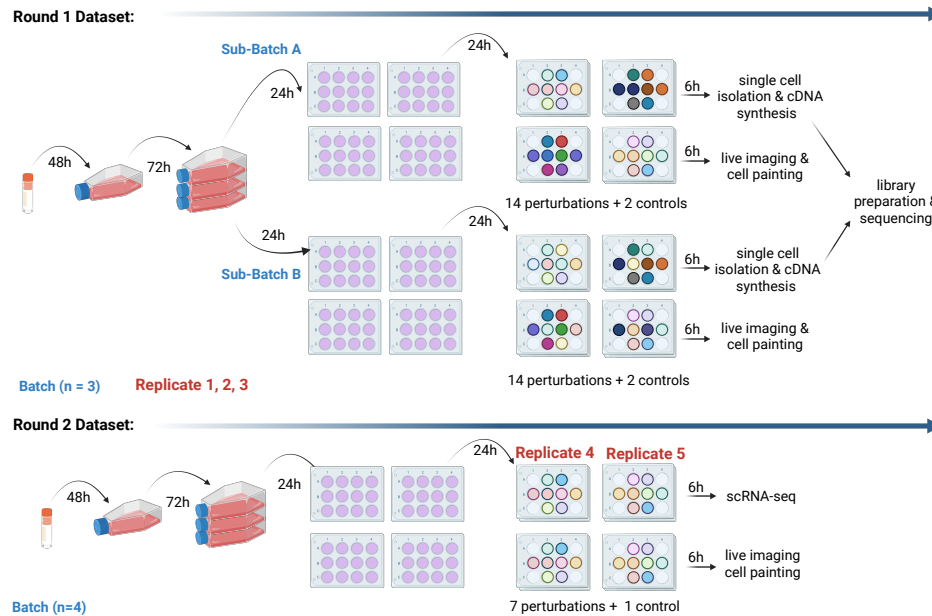


Figure 3: Illustration of the sequential steps involved in experimental data generation rounds 1 and 2 resulting in the full scGeneScope dataset. Multimodal datasets were created by pairing scRNA-seq with Cell Painting of U2-OS cells, which were perturbed by 28 chemicals and one solvent control (DMSO). A total of five sets of replicates per condition were collected, divided between the two experimental data generation rounds (three sets of replicates for Round 1 and two sets of replicates for Round 2). The Round 1 dataset is comprised of three batches, with each batch representing a replicate that includes all 28 unique perturbations. These batches were processed as two *sub-batches* on the same day (sub-batches A and B), each containing 14 perturbations. For scRNA-seq library preparation and sequencing, samples from all Round 1 batches were collected and processed simultaneously. The Round 2 dataset was generated in four batches, each consisting of two sets of replicates of seven unique perturbations and one solvent control (DMSO) per plate. Unlike Round 1, the batches were processed completely separately. In all datasets, the plate maps were scrambled between replicates but remained consistent between plates for imaging and scRNA-seq. To ensure tight pairing of multimodal data, plates for each measurement (imaging and scRNA-seq) were processed side by side.

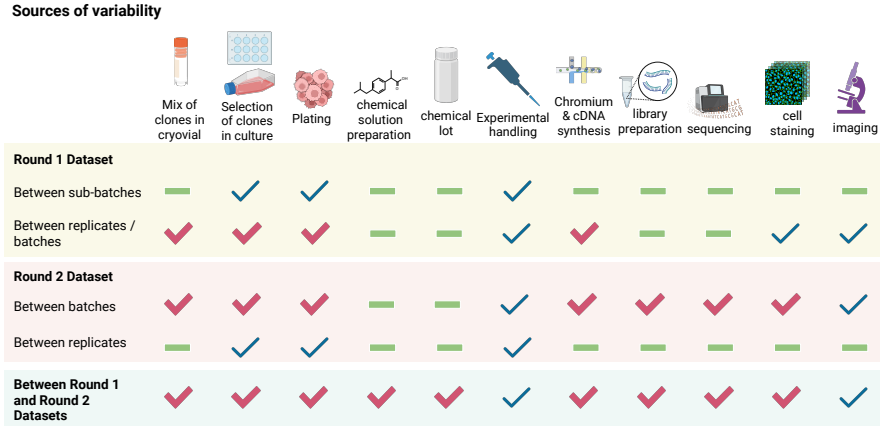


Figure 4: **Schematic representing sources of variability between sub-batches, batches, replicates, and datasets and their relative significance.** A bold red check mark indicates a significant source of variability, a thin blue check mark indicates a low-moderate source of variability, and a green minus indicates a likely insignificant contribution to experimental variability. Potential sources of variation include the mix of clones in cryovial, the selection of clones in culture, the heterogeneity in cell plating and cell count, the chemical solution preparation, the chemical lot, experimental handling of samples, the chromium run and cDNA synthesis, the library preparation, sequencing, cell staining, and imaging. These sources of variability could impact measurements between sub-batches, batches and replicates of the same experimental round, and between experimental rounds.

367 differences between biological replicates and technical replicates. Here, we clarify the nature of
 368 biological and technical replicates in our experimental process.

369 During the data generation process for the scGeneScope dataset, there were two independent exper-
 370 imental data generation rounds with experimental procedures depicted in Figure 3 and detailed in
 371 Section A.1. During Round 1, there were three batches to yield three sets of biological replicates.
 372 For each biological replicate, a single flask of U2-OS cells was cultured and subsequently used to
 373 seed individual wells on eight 12-well plates. Working in two sub-batches of 14 treatments, all
 374 28 individual treatments were then applied a total of two times each to individual well, yielding
 375 two technical replicates per treatment. Per treatment, one technical replicate was assayed with Cell
 376 Painting microscopy, and the other technical replicate was assayed by single-cell RNA sequencing.
 377 During generation Round 2, data was generated through four consecutive batches. In each batch,
 378 a single flask of U2-OS cells was grown up and split onto four 12-well plates. Batches of seven
 379 individual treatments were then applied a total of four times to different wells, yielding four tech-
 380 nical replicates per treatment. Two of these technical replicates were measured with Cell Painting
 381 microscopy, and two technical replicates were measured with single-cell RNA sequencing. We note
 382 that in data generation Round 2, because treatments are grouped into batches of seven and done in
 383 consecutive waves, treatment and batch are confounded, making this round of replicates unsafe to
 384 split across train and test sets. Generation Round 1 does not have the same confounded variables and
 385 thus is safe to split across train and test sets.

386 A.1.2 Chemical Selection

387 To create the scGeneScope dataset, we started by identifying a set of chemical treatments with unique
388 targets expressed in U2-OS cells, with a diverse range of phenotypic responses across our target
389 modalities. Our approach combined data-driven analyses in gene expression and Cell Painting (CP)
390 imaging spaces and perturbation nomination driven by expert biological knowledge. The integration
391 of these analyses provided a guide to nominate candidate perturbations.

392 We employed a two-step strategy to assemble a final panel of 28 small-molecule treatments used to
393 create the scGeneScope dataset. We first leveraged a data-agnostic approach, where an existing U2-
394 OS Cell Painting dataset was mined [46] for annotations that mapped compounds to their molecular
395 targets. To maximize biological breadth, 14 non-overlapping targets were chosen, followed by
396 nomination of one well-behaved chemical compound per target, prioritizing acceptable toxicity,
397 solubility, assay robustness, and availability of appropriate controls.

398 Next, we expanded the panel by integrating data-driven insights from the Rosetta L1000 datasets [5]
399 as well as from the CP-JUMP [46] library. The objective of the Rosetta L1000 analysis was to
400 rank compounds across the LINCS and CDRP-Bio chemical perturbation datasets, which include
401 screens against the U2-OS osteosarcoma cell line. Well-level L1000 measurements were downloaded,
402 gene expression vectors were standardized and filtered, and the Pearson correlation coefficients
403 between replicates were computed. Compounds with high replicate correlation were retained, and
404 gene expression data was re-standardized. Compound effects were quantified via Euclidean distance
405 between the gene expression vector for compound-treated cells and the gene expression vector for the
406 on-plate control. Compounds were ranked by this distance metric to guide downstream nomination.

407 The CP-JUMP analysis aimed to rank and nominate compounds based on phenotypic effects measured
408 by Cell Painting imaging. Euclidean distances in CellProfiler feature space were computed between
409 compound-treated wells and control wells in CellProfiler feature space. CellProfiler features were
410 projected to 50 principal component analysis (PCA) dimensions, and distances were standardized to
411 account for batch effects. Compounds that yielded reproducible results in at least four sources were
412 retained. This process yielded a list of 350 compounds, ranked by their relative distanced induced,
413 averaged across sources.

414 Seven compounds total were nominated the Rosetta analysis and seven from the CP-JUMP analyses,
415 based on magnitude of phenotypic effect induced and orthogonality with the compounds nominated
416 via biological and chemical prioritization. This process yielded a final list of 28 small molecule
417 compounds (Table 2).

418 Finally, for all 28 compounds, we confirmed expression of target genes in untreated U2-OS cells, and
419 an unambiguous target annotation, favoring distinct targets over strict mechanism-of-action labels
420 to avoid sparsity. The resulting set of 28 perturbations provides broad, non-redundant coverage of
421 cellular pathways while remaining compatible with routine Cell Painting and single-cell RNA-seq
422 assays.

423 A.1.3 Biological Materials and Methods

424 **Cell Culture** U2-OS cells (ATCC #HTB-96) were cultured in McCoy's 5a Medium Modified
425 (Gibco #16600-082) with 10% FBS (Omega #FB-01) in 37° C at 5% CO₂. Cells were plated in
426 12-well glass bottom plates (CellVis #P12-1.5H-N) 24 hours before the start of treatment in exact
427 duplicate plates. Compounds were added to the wells in the doses listed in Table 2 in a final solution
428 of 0.1% v/v dimethyl sulfoxide (Thermo Scientific Chemicals #J66650AP), with the location of
429 compounds shuffled between consecutive rounds of replicates. Dosages were selected for each
430 treatment by reviewing the dosages used in prior works [46, 2], and selecting ranges with low levels
431 of toxicity. After 6 hours of treatment, plates were processed for scRNA-seq and Cell Painting.

432 **Cell Painting** After 5.5 hours of incubation with indicated treatments, Mitotracker Deep Red FM
433 (CST #8778) was spiked into each well for a final concentration of 500 nM and incubated for 30
434 min. Cells were fixed in freshly diluted 3.2% v/v paraformaldehyde (EM Sciences #50980495) at
435 room temperature for 20 min, washed three times with 1x HBSS (Gibco #14-175-103), and stained
436 with Cell Painting staining mix (6 μ M Syto14 (Fisher #S7576), 1.25 μ L/mL Phalloidin AlexaFluor
437 568 (Fisher #A12380), 50 μ g/mL Concanavalin A CF750 (Biotium #29080), 1.5 μ g/mL WGA568
438 (Biotium #29077), 2 μ g/mL Hoechst 34580 (Sigma #63493), and 1x PhenoVue™ Dye Diluent A

Table 2: Chemical treatments used in this study, with primary target and mechanism of action (MoA) [2], supplier, CAS number, and dosage information.

Chemical	Primary target / MoA	Vendor	Cat.#/CAS	Dose
Phenacetin	Cyclooxygenase inhibitor	Cayman	62-44-2	11.16 μ M
PQ401	IGF-1 / IGF-1R inhibitor	Cayman	196868-63-0	10 μ M
Splitomicin	SIRT inhibitor	Cayman	138-433-9	22.22 μ M
(R) -MG132	Proteasome inhibitor	Cayman	1211877-36-9	10.51 μ M
(R) -Roscovitine	CDK inhibitor	Cayman	186692-46-6	14.11 μ M
Wy 14643 / Pirinixic Acid	PPAR receptor agonist	Cayman	50892-23-4	10 μ M
Fluocinonide	Glucocorticoid receptor agonist	Cayman	0356-12-7	4.04 μ M
Caffeine	Adenosine receptor antagonist	Cayman	58-08-2	10 μ M
LY303511 (hydrochloride)	Casein-kinase / mTOR / PI3K inhibitor	Cayman	854127-90-5	10 μ M
Simvastatin	HMG-CoA-reductase inhibitor	Cayman	79902-63-9	10 μ M
Colchicine	Microtubule inhibitor	Cayman	64-86-8	10 μ M
Pantoprazole	ATPase inhibitor	Cayman	102625-70-7	10 μ M
Benzbromarone	Chloride-channel blocker	Cayman	3562-84-3	4.72 μ M
AMG-900	Aurora-kinase inhibitor	Cayman	945595-80-2	10 μ M
DBeQ	p97 inhibitor	Cayman	177355-84-9	10 μ M
Daporinad / FK-866	Transferase inhibitor	Cayman	658084-64-1	10 μ M
Vorinostat / SAHA	Histone-deacetylase inhibitor	Cayman	149647-78-9	10 μ M
Quinidine	Sodium-channel blocker	Cayman	56-54-2	10 μ M
Aloxistatin / E-64d	Cysteine-protease inhibitor	Cayman	88321-09-09	10 μ M
Cycloheximide	Protein-synthesis inhibitor	Cayman	66-81-9	10 μ M
Thapsigargin	SERCA inhibitor	Cayman	67526-95-8	7.68 μ M
BAY 11-7082	NF- κ B-pathway inhibitor	Cayman	19542-67-7	24.14 μ M
CGK-733	ATM/ATR-kinase inhibitor	Cayman	905973-89-9	10 μ M
PD-98059	MEK / MAP-kinase inhibitor	Cayman	167869-21-8	18.71 μ M
GW-843682X	PLK inhibitor	Cayman	660868-91-7	10 μ M
PMA	PKC activator	Cayman	16561-29-8	3.06 μ M
SKI-II	Sphingosine-kinase inhibitor	Cayman	312636-16-1	10 μ M
HARMAN	Monoamine-oxidase inhibitor	Cayman	486-84-0	10 μ M

(Perkin Elmer LLC #50-209-3540) in 0.1% v/v Triton-X100 (Fisher BioReagents #BP151-500)) for 30 min at room temperature in the dark. After washing three times with 1x HBSS, cells were imaged using Nikon Ti2E with Perfect Focus and 25mm Field of View (FOV) with a CFI PLAN APO LAMBDA 40x, NA 0.95 objective, and C-FL epifluorescence cubes.

Single-cell RNA Sequencing Single-cell RNA-sequencing (scRNA-seq) was performed in batches of 16 samples. Briefly, cells were lifted after incubation with TrypLE express (Gibco #12604039) at 37°C 5%CO₂ for 5 min. After determining the cell concentration for each sample using Vi-CELL BLU Cell Viability Analyzer (Beckmann Coulter), the cells were normalized to 1000 cells/ μ l in 1x DPBS (Gibco # 14-190-250) + 0.04% w/v BSA (Sigma #A1595). Single cell suspensions were then processed with the 10x Genomics Platform according to manufacturers' instructions using Chromium Next GEM Single Cell 3' Kit v3.1 reagents for a targeted recovery of 10,000 cells per sample. Libraries were sequenced using Illumina's NovaSeq X Plus on a 25B flow cell with the following sequencing parameters: 28/10/10/90.

A.1.4 Data Preprocessing

Image Processing Cell Painting images were of shape 2250 pixels x 2250 pixels x 13 z-stack slices, with 6 images acquired, including a brightfield channel, as well as stain channels for Wheat Germ Agglutinin CF568 (Actin, Golgi, and Plasma membrane | AGP), Concanavalin A CF750 (Endoplasmic Reticulum), MitoTracker DeepRedFM (Mitochondria), Hoechst 33342 (DNA), Syto14 (RNA) according the Cell Painting protocol [1]. From the 13 z-stacks, the final focal plane was calculated as follows using the DNA channel: each slice in the z-stack was Gaussian blurred with a (3,3) kernel and convolved with a discrete Laplacian operator, with the variance of the resulting image calculated. The slice with maximal variance in the DNA channel was designated as the focal plane across all channels for the given sample [47]. Nuclei were subsequently segmented using CellPose 3.1.1.1 using just the focal plane of the DNA channel with the following model parameters and hyperparameters: nuclei model, flow threshold of 0.8, diameter of 100 pixels [48]. CellPose

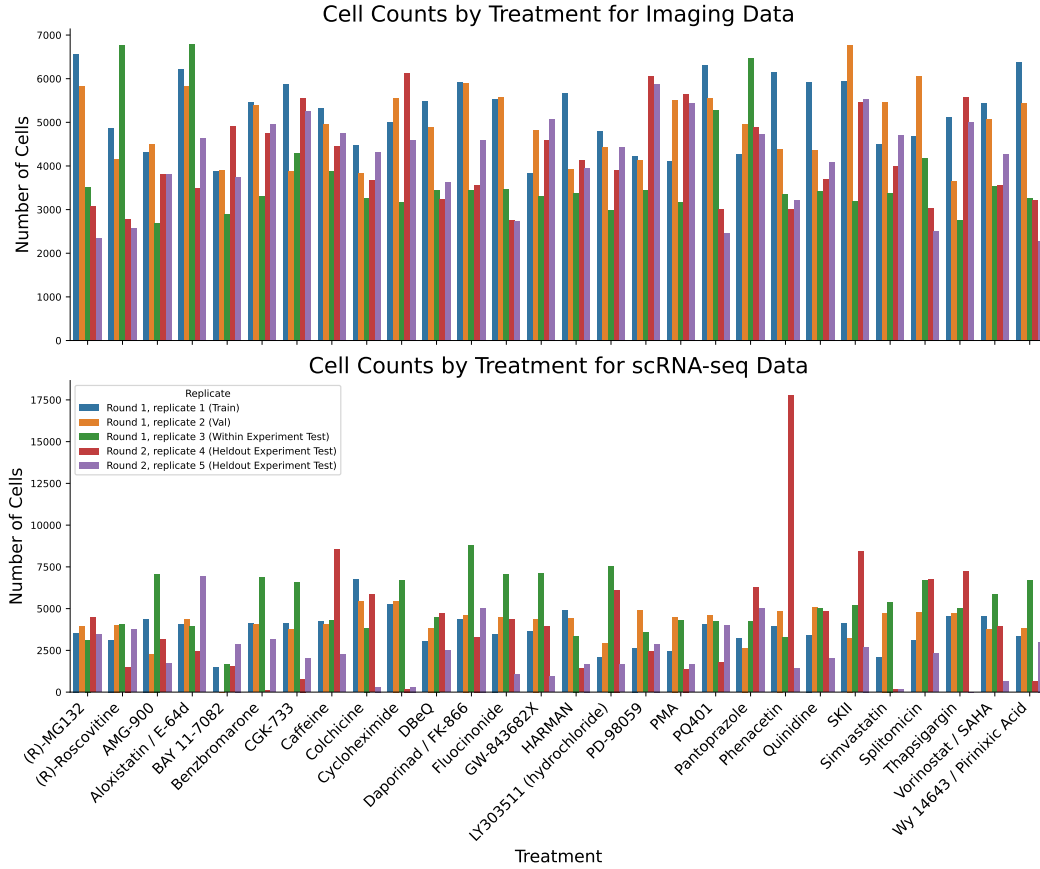


Figure 5: Single-cell profile counts plotted by treatment and replicate for Cell Painting and scRNA-seq modalities.

segmentation outputs were then post-processed such that only nuclei with an area between 1,000 and 100,000 pixels that were at least 100 pixels from the edge of the acquired field of view were kept. Box edge coordinates that passed these criteria were saved, ultimately yielding single cell image patches.

At inference time, focal plane images were loaded and each channel was processed separately: clipping at [0.05,99.95] percentiles, standardization using channel-wise, dataset-wide means and standard deviation, and min-max normalization. Nuclei-centered boxes of size 224 x 224 were then cropped from the focal plane images and used for model inference.

scRNA-seq Processing scRNA-seq data was processed on the Seqera Platform with standard parameters using the nf-core scRNA-seq pipeline version 2.5.0, with 10XV3 chemistry, and no prebuilt index. Postprocessing was completed using the output alevin mtx conversions in R. Cells were further filtered for high quality to only those containing a minimum of 2500 genes and 4000 total counts, and with less than 15% mitochondrial RNA. Where necessary, EnsDb.Hsapiens.v86 was used to map gene symbols to Ensembl IDs.

A.2 Additional Modeling Details

A.2.1 Generating Image Embeddings

Embeddings were generated from ImageNet-pretrained models using nuclei-centered patches processed using details in the Image Processing section. ResNet50 and ResNet152 architectures were loaded with IMAGENET1K_V2 pretrained weights, while ViT-L-16 and ViT-H-14 were loaded with IMAGENET1K_SWAG_LINEAR_V1 pretrained weights. Both ResNet architectures were used as feature extractors by extracting embeddings after the final "flatten" layer, while both ViT

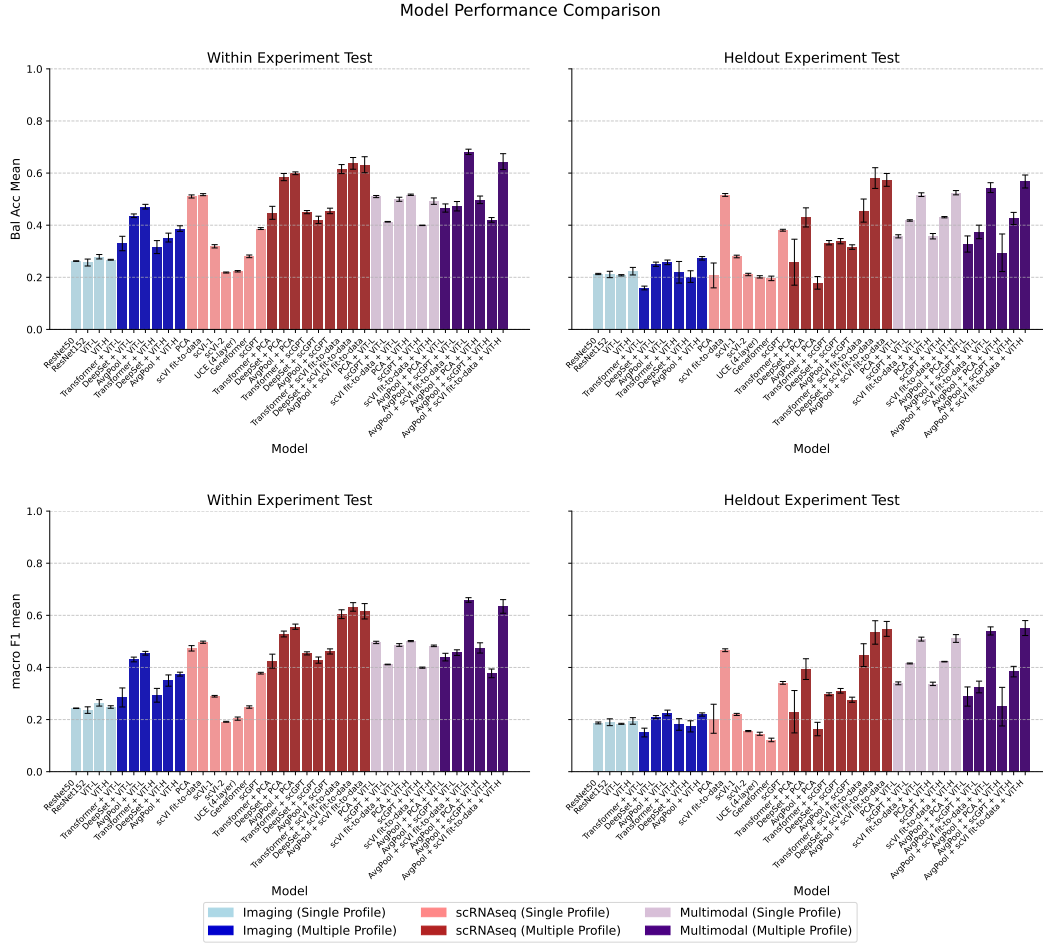


Figure 6: **Visualization of all results.** All results from Table 1 are visualized; the left column shows Within Experiment (WE) evaluations and the right column shows Heldout Experiment (HE) evaluations, while the top row shows Balanced Accuracy and the bottom row shows Macro F1 scores.

architectures were converted into feature extractors by looking at embeddings after the "getitem_5" layer. ViT-L-16 and ViT-H-14 were used as feature extractors by taking the average of all image patch tokens from the last transformer layer. ResNet50 and ResNet152 embeddings were of size 2048, ViT-L-16 embeddings were of size 1024, and ViT-H-14 embeddings were of size 1280. Embeddings were generated for each Cell Painting stain independently, triplicating each single channel stain image into the 3-channel inputs required for ImageNet-pretrained models. Finally the embeddings for each stain were concatenated together together, giving total scImage embedding sizes of 10,240 for ResNet50 and ResNet152, and 5,120 and 6,140 for ViT-L-16 and ViT-H-14 respectively.

A.2.2 Generating scRNA-seq Embeddings

We followed the recommended protocol for each model to ensure best practices in embedding generation. In all cases, raw counts were used to generate embeddings. Batch information was incorporated when the model supported its use. For scGPT and scVI (fit to train), highly variable genes (HVGs) were selected as part of the data preprocessing. HVGs were computed on the training split only, using scanpy with `flavor=seurat_v3`, and applied to all other splits so as to avoid data leakage. For PCA (fit to train) and scVI (fit to train), only the training replicates were used to fit the model, and embeddings for all other replicates, including validation and testing replicates, were obtained using the fitted model. For the pretrained foundational models (scVI-1, scVI-2, UCE, Geneformer, and scGPT), the pretrained models were downloaded and directly used for embedding generation according to the respective documentation.

Table 3: Unimodal single profile models hyperparameter ranges.

Hyperparameter	Distribution
<i>Hidden dimension size:</i>	
hidden_dim	Int: 256 to 4096, log=True
<i>Number of layers in the classifier:</i>	
depth	Int: 1 to 7
<i>Learning rate:</i>	
lr	Float: 1e-5 to 1e-3, log=True
<i>Weight decay (L2 penalty) for the optimizer:</i>	
weight_decay	Float: 1e-6 to 1e-2, log=True
<i>Dropout rate:</i>	
dropout_rate	Categorical: [0, 0.25, 0.5]

Table 4: Multimodal single profile models hyperparameter ranges.

Hyperparameter	Distribution
<i>Joint classifier hidden dimension size:</i>	
hidden_dim	Int: 256 to 4096, log=True
<i>Number of layers in the joint classifier:</i>	
depth	Int: 1 to 7, step=2
<i>Learning rate:</i>	
lr	Float: 1e-5 to 1e-3, log=True
<i>Weight decay (L2 penalty) for the optimizer:</i>	
weight_decay	Float: 1e-6 to 1e-2, log=True
<i>Dropout rate:</i>	
hidden_dropout	Categorical: [0, 0.25, 0.5]
<i>RNA-seq encoder hidden dimension size:</i>	
rnaseq.hidden_dim	Int: 256 to 4096, log=True
<i>RNA-seq encoder number of layers:</i>	
rnaseq.depth	Int: 1 to 7, step=2
<i>RNA-seq encoder dropout rate:</i>	
input_dropout.rnaseq	Categorical: [0, 0.25, 0.5]
<i>Imaging encoder hidden dimension size:</i>	
imaging.hidden_dim	Int: 256 to 4096, log=True
<i>Imaging encoder number of layers:</i>	
imaging.depth	Int: 1 to 7, step=2
<i>Imaging encoder dropout rate:</i>	
input_dropout.imaging	Categorical: [0, 0.25, 0.5]
<i>Output dimension per modality:</i>	
OUTPUT_DIM_PER_MODALITY	Int: 256 to 4096, log=True

503 A.2.3 Fitting Task Heads

504 The task heads are designed as a series of PyTorch operations mapping the high-dimensional fea-
505 ture space to the target output space. The architecture is defined within the model configuration,
506 leveraging Hydra’s configuration management capabilities to ensure flexibility and reproducibility.
507 Data preprocessing is managed by a dedicated data module, implemented using PyTorch Lightning’s
508 LightningDataModule. This module is responsible for loading, augmenting, and batching the
509 data, ensuring that it is in the appropriate format for the model. The data module’s configuration is

Table 5: Unimodal AvgPool multi-profile models hyperparameter ranges.

Hyperparameter	Distribution
<i>Classifier hidden dimension size:</i>	
hidden_dim	Int: 256 to 4096, log=True
<i>Number of layers in the classifier:</i>	
depth	Int: 1 to 7
<i>Learning rate:</i>	
lr	Float: 1e-5 to 1e-3, log=True
<i>Weight decay (L2 penalty) for the optimizer:</i>	
weight_decay	Float: 1e-6 to 1e-2, log=True
<i>Dropout rate:</i>	
dropout_rate	Categorical: [0, 0.25, 0.5]

Table 6: Unimodal DeepSet multi-profile models hyperparameter ranges.

Hyperparameter	Distribution
<i>Encoder elementwise depth:</i>	
encoder.elementwise_depth	Int: 1 to 3
<i>Number of layers in the encoder:</i>	
encoder.num_layers	Int: 1 to 3
<i>Number of layers in the classifier:</i>	
classifier.depth	Int: 1 to 7
<i>Learning rate:</i>	
lr	Float: 1e-5 to 1e-3, log=True
<i>Weight decay (L2 penalty) for the optimizer:</i>	
weight_decay	Float: 1e-6 to 1e-2, log=True
<i>Encoder and classifier hidden dimension size:</i>	
LATENT_DIM	Int: 256 to 1024, log=True

Table 7: Unimodal Transformer multi-profile models hyperparameter ranges.

Hyperparameter	Distribution
<i>Number of heads in the transformer:</i>	
encoder.n_heads	Categorical: [1, 4, 8]
<i>Feedforward dimension in the transformer:</i>	
encoder.dim_feedforward	Categorical: [256, 512, 1024]
<i>Dropout rate in the transformer:</i>	
encoder.dropout	Categorical: [0, 0.1, 0.25]
<i>Number of layers in the transformer:</i>	
encoder.num_layers	Categorical: [1, 2, 4]
<i>Learning rate:</i>	
lr	Float: 1e-5 to 1e-3, log=True
<i>Weight decay (L2 penalty) for the optimizer:</i>	
weight_decay	Float: 1e-6 to 1e-2, log=True

510 specified in `cfg.data` (see Section A.4.2 for further details), allowing for seamless integration with
511 the training pipeline. The training of task heads is conducted using PyTorch Lightning’s `Trainer`
512 class. This process is augmented by a suite of callbacks that facilitate early stopping, learning

Table 8: Multimodal multi-profile models hyperparameter ranges.

Hyperparameter	Distribution
<i>Joint classifier hidden dimension size:</i>	
classifier.hidden_dim	Int: 256 to 4096, log=True
<i>Number of layers in the joint classifier:</i>	
classifier.depth	Int: 1 to 7, step=2
<i>Learning rate:</i>	
lr	Float: 1e-5 to 1e-3, log=True
<i>Weight decay (L2 penalty) for the optimizer:</i>	
weight_decay	Float: 1e-6 to 1e-2, log=True
<i>Dropout rate:</i>	
hidden_dropout	Categorical: [0, 0.25, 0.5]
<i>RNA-seq encoder hidden dimension size:</i>	
RNASEQ_HIDDEN_DIM	Int: 256 to 4096, log=True
<i>RNA-seq encoder number of layers:</i>	
RNASEQ_DEPTH	Int: 1 to 7, step=2
<i>RNA-seq encoder dropout rate:</i>	
input_dropout.rnaseq	Categorical: [0, 0.25, 0.5]
<i>Imaging encoder hidden dimension size:</i>	
IMAGING_HIDDEN_DIM	Int: 256 to 4096, log=True
<i>Imaging encoder number of layers:</i>	
IMAGING_DEPTH	Int: 1 to 7, step=2
<i>Imaging encoder dropout rate:</i>	
input_dropout.imaging	Categorical: [0, 0.25, 0.5]
<i>Output dimension per modality:</i>	
OUTPUT_DIM_PER_MODALITY	Int: 256 to 4096, log=True

rate scheduling, and checkpointing. To ensure the reproducibility of results, a fixed random seed is employed across all experiments. This is achieved using the `Lightning.seed_everything` function, which synchronizes the random number generators across PyTorch, NumPy, and Python’s random module.

We ran hyperparameter optimization for each model in each setting separately. After setting the best hyperparameters, each model is trained with five different seeds to report final metrics. We have used `optuna` library for hyperparameter optimization. Thus, we define all hyperparameter ranges using `optuna` distributions, which can be categorized as `categorical`, `int`, or `float`. The hyperparameter ranges and their corresponding distribution classes, are detailed in tables 3 to 8.

A.3 Compute

All training runs were conducted on AWS `g6.4xlarge` instances, which are part of the Amazon EC2 G6 instance family. Each `g6.4xlarge` instance is equipped with NVIDIA L4 Tensor Core GPUs, powered by 16 vCPUs, and comes with 64 GiB of RAM. RAM and CPU usage is maximized by maxing out `num_workers` for the dataloader. The training run times varied significantly depending on the complexity and configuration of the models being trained. For the simpler unimodal single profile runs, run times were as short as one minute. In contrast, the more complex multimodal multi-profile runs had run times extending up to six hours.

For hyperparameter optimization runs, we ran up to 300 trials for unimodal models, and up to 800 trials for the multimodal models using `optuna`’s `TPESampler` sampling strategy. In total, for these experiments we have used approximately 24,000 GPU hours.

533 A.4 Software Library

534 The scGeneScope library provides a comprehensive framework for multimodal and multiprofile
535 integration in single-cell phenotypic profiling. Built on modern Python machine learning frameworks
536 (PyTorch, PyTorch Lightning, and Hydra), it offers a modular and extensible architecture that
537 enables rapid development and evaluation of novel computational methods. The library emphasizes
538 three key design principles: (1) ease of use through clear abstractions and minimal boilerplate code,
539 (2) flexibility to accommodate diverse model architectures and data types, and (3) reproducibility via
540 standardized training and evaluation pipelines. Researchers can access the open-source implementa-
541 tion at <https://github.com/altos-labs/scGeneScope>. Next, we provide a detailed exposition
542 of the software architecture and core abstractions.

543 A.4.1 Frameworks

544 scGeneScope is built upon widely-adopted machine learning and single-cell analysis frameworks,
545 carefully chosen to maximize accessibility and maintainability. By utilizing established libraries that
546 are standard in both the machine learning and computational biology communities, we reduce the
547 barrier to entry for researchers wanting to use or extend our benchmark suite. These foundational
548 frameworks provide well-documented patterns and architectural guidelines that we leverage to create
549 a robust and intuitive codebase structure. The following sections detail the key frameworks underlying
550 scGeneScope and their specific roles in our implementation.

551 **PyTorch:** Serving as our core deep learning framework, PyTorch [49] provides efficient auto-
552 matic differentiation and neural network primitives through a dynamic computational graph ap-
553 proach. We leverage PyTorch’s comprehensive ecosystem for building and optimizing neural
554 architectures, with particular emphasis on its data handling abstractions. Specifically, we utilize
555 the `torch.utils.data.Dataset` and `torch.utils.data.DataLoader` interfaces to implement
556 our data processing pipeline in the `scgenescope.data` module, enabling efficient data loading and
557 batching during model training.

558 **PyTorch Lightning:** To enhance code clarity and reduce boilerplate while maintaining flexibility,
559 we employ PyTorch Lightning [50] as our high-level training framework. Lightning provides struc-
560 tured abstractions that separate model logic from training mechanics through its `LightningModule`
561 and `LightningDataModule` interfaces. We encapsulate our models and datasets within these classes,
562 allowing Lightning’s `Trainer` to handle training loop logistics, device management, and distributed
563 training seamlessly. Additionally, Lightning’s callback system facilitates integration with monitoring
564 tools like TensorBoard, enabling comprehensive experiment tracking.

565 **Hydra:** Complex machine learning pipelines require robust configuration management. We utilize
566 Hydra [51] to provide a hierarchical configuration system that elegantly handles the complexity of
567 our benchmark suite. Hydra enables composition-based configuration management, where different
568 components can be configured independently and combined as needed. This approach facilitates
569 reproducible experiments through comprehensive configuration logging, while providing conven-
570 ient features such as command-line completion, hyperparameter optimization via Optuna, and
571 configuration validation through type checking.

572 **AnnData:** For biological data management, we adopt AnnData [52] as our primary data structure.
573 AnnData provides a specialized container for single-cell RNA sequencing data, storing both the gene
574 expression matrix and associated metadata. Each AnnData object maintains cell-level annotations
575 (including perturbation information and covariates) and gene-level metadata (such as gene identifiers).
576 Our data module interfaces with AnnData through the `h5ad` file format, while our analysis module
577 generates predictions in AnnData format, ensuring compatibility with the broader single-cell analysis
578 ecosystem.

579 A.4.2 Data Abstractions

580 Our dataset presents unique challenges for data management and model training. Each treatment
581 condition contains two large collections of single-cell profiles - one for scRNA-seq and one for
582 scImage data. Due to memory constraints, training on complete cell populations is infeasible, neces-
583 sitating intelligent sampling strategies. To address these challenges, we developed a hierarchical data

management system built on three core abstractions: `Population` (Listing 2), `AlignedPopulation` (Listing 3), and `SampledPopulation` (Listing 4). This system enables flexible data loading, consistent sampling, and reproducible experimentation.

Single Cell Profile: The foundation of our data management system is the `Samples` class, which extends the base `DataStore` class to provide a unified interface for single-cell measurements (Listing 1). Each `DataStore` maintains a table of cell profiles and their associated metadata (e.g., treatment conditions, covariates) through its key attributes:

- `store`: An indexed collection of cell profile data, storing the actual measurements in a format that supports efficient access and iteration
- `observations`: A pandas `DataFrame` containing metadata and annotations for each cell profile, such as treatment conditions, batch information, and other experimental covariates

The `Samples` class augments this with standardized methods for reading common biological data formats like `AnnData`, `HDF5`, and `CSV` through class methods:

- `from_anndata`: Creates a `Samples` instance from an `AnnData` object, preserving both expression data and metadata
- `from_embeddings`: Initializes a `Samples` instance from pre-computed embeddings or feature representations

Both classes implement PyTorch’s `Dataset` interface, enabling seamless integration with existing data loading utilities.

```

603 1 @dataclass
604 2 class DataStore(torch.utils.data.Dataset, Generic[T]):
605 3     """A tabular single-cell dataset."""
606 4     store: Indexed[T]
607 5     observations: pd.DataFrame | None
608 6
609 7 @dataclass
610 8 class Samples(DataStore):
611 9     """An interface for a single-cell profile dataset."""
61210     @classmethod
61311     def from_anndata(cls, ...):
61412         ...
61513     @classmethod
61614     def from_embeddings(cls, ...):
61715         ...

```

Listing 1: Data structure representing the single cell data.

Multi-Profile Data: The `Population` class extends `Samples` to handle grouped single-modality data (Listing 2). It provides a structured view of cell profiles through a dictionary of grouped conditions (e.g., treatments), supporting both integer-based and string-based indexing for flexible data access. This abstraction simplifies the organization and retrieval of related cell profiles while maintaining the underlying data structure. The class has several key attributes:

- `samples`: The underlying data store containing the single-cell profiles, implementing the `SupportsGrouping` interface for grouped access
- `condition_on`: The column name in the metadata used to group the samples (e.g., "treatment" or "cell_type")
- `indexing`: Controls how samples are indexed, supporting both label-based (`CONDITION`) and position-based (`LOC`) access
- `transform`: An optional function applied to samples when accessed, enabling on-the-fly data transformations
- `include_condition`: When `True`, includes the condition label alongside the sample data

- grouped: A mapping from condition labels to sequences of sample indices, providing the core grouping functionality

```

634 | @dataclass
635 | class Population:
636 |     """A dataset of single-cell profiles grouped by a condition."""
637 |     samples: SupportsGrouping
638 |     condition_on: str
639 |     indexing: Index = Index.LOC
640 |     transform: Callable | None = None
641 |     include_condition: bool = False
642 |     grouped: Mapping[str, Sequence[int]]

```

Listing 2: Pytorch dataset classes for training.

Multi-Modal Data: For integrating multiple data modalities, we developed the ConditionAlignedPopulation class (Listing 3). This abstraction aligns different modalities (e.g., scRNA-seq and imaging data) based on shared experimental conditions through its join_on_condition method. The join_on_condition method takes multiple dataset factory functions and a condition name as input, and creates aligned populations by matching samples across modalities that share the same condition values. For example, given RNA-seq and imaging datasets with matching treatment labels, this method would align cells from both modalities that received the same treatment, enabling joint analysis while preserving the individual characteristics of each data type. The method handles the complexity of index matching and data alignment internally, providing a clean interface for working with multi-modal data.

```

653 | class ConditionAlignedPopulations(torch.utils.data.StackDataset):
654 |     """A dataset of condition-aligned populations."""
655 |     ...
656 |     @classmethod
657 |     def join_on_condition(
658 |         cls,
659 |         *dataset_factories: Callable[[], SupportsIndexModeSelect],
660 |         condition: str,
661 |         ...
662 |     ) -> Self:
663 |         ...

```

Listing 3: Condition aligned dataset.

Sampled Population: To address computational constraints, the SampledPopulation class implements controlled down-sampling of cell populations (Listing 4). It supports both direct specification of sampling strategies via custom samplers and simplified sampling through numerical parameters. This abstraction ensures reproducible sub-sampling while maintaining statistical validity of the resulting datasets. The class has two key attributes:

- num_samples: Controls the size of the sampled population. Can be specified as a single integer for fixed-size sampling, a tuple of (min, max) for random-size sampling, or None to use the full population.
- sampler: A callable function that implements the sampling strategy. Takes a list of indices and returns a subsampled list. When None, uses uniform random sampling based on num_samples.

```

675 | @dataclass(kw_only=True)
676 | class SampledPopulation(Population):
677 |     """A downsampled population dataset."""
678 |     num_samples: int | tuple[int, int] | None = None
679 |     sampler: Callable[[list[int]], list[int]] | None = None

```

Listing 4: Sampled population class.

680 **Data Pipeline Configuration:** Each of the above classes is used in our data configuration system
681 given below. Please refer to individual files for how to specifically configure the data classes.

```
data/
├── source/
│   └── embeddings/
├── iterators/
│   ├── aligned_iterables.yaml
│   ├── conditional_alignment.yaml
│   ├── iterable_populations.yaml
│   ├── make_iterable.yaml
│   ├── samples.yaml
│   └── sampled_populations.yaml
├── pipeline/
│   ├── multimodal/
│   │   └── rna_imaging_multiprofile.yaml
│   ├── multiprofile/
│   │   ├── multipleinput.yaml
│   │   └── singleinput.yaml
│   ├── singleprofile/
│   │   ├── multipleinputs.yaml
│   │   └── singleinput.yaml
│   └── transform/
│       ├── multiple_input_sample_transform.yaml
│       └── treatment/
```

682 A.4.3 Reproducibility and Deterministic Datasets

683 To ensure reproducibility while maintaining dataset diversity, we implement a careful balance between
684 deterministic and stochastic sampling approaches. For single-cell profile datasets, determinism is
685 inherent as the profiles are fixed. For multi-profile and multi-modal datasets, we employ controlled
686 stochastic sampling during training to create diverse treatment response datasets, while enforcing
687 reproducibility during validation and testing through specialized seed-controlled iterators. This design
688 enables rigorous benchmarking while preserving the benefits of data augmentation during the training
689 phase.

690 A.4.4 Model Abstractions

691 To enable systematic evaluation of treatment response prediction approaches, we provide two core
692 model abstractions: `UnimodalSampleClassifier` and `MultiModalMultipleInputClassifier`.
693 Both inherit from `LitClassifier`, which extends PyTorch Lightning's `LightningModule` to pro-
694 vide standardized training, validation, and evaluation interfaces.

695 **Unimodal Classification:** The `UnimodalSampleClassifier` (Listing 5) implements single-
696 modality treatment response prediction. It consists of:

- 697 • **encoder:** A neural network module that projects input data into a latent representation
- 698 • **classifier:** A module that maps the latent representation to treatment predictions
- 699 • **optimizer_factory:** A callable that constructs the optimizer for model parameters
- 700 • **scheduler_factory:** A callable that creates the learning rate scheduler
- 701 • **compile:** A boolean flag enabling PyTorch 2.0 compilation for improved performance

```
702 1 class UnimodalSampleClassifier(LitClassifier):
703 2     """A unimodal classifier.
704 3
705 4     This model takes a single modality as input and outputs a
706 5     prediction.
707 6
708 7     Attributes:
709 8         encoder: The encoder model.
```

```

710 8         classifier: The classifier model.
711 9         """
71210
71311     def __init__(
71412         self,
71513         encoder: torch.nn.Module | None,
71614         classifier: torch.nn.Module,
71715         optimizer_factory: Callable[
71816             [Iterable[torch.nn.Parameter]], torch.optim.Optimizer
71917         ],
72018         scheduler_factory: Callable[[torch.optim.Optimizer], torch.
721optim.lr_scheduler],
72219         compile: bool,
72320     ) -> None:

```

Listing 5: Unimodal classifier implementation.

724 **Multi-Modal Classification:** The `MultiModalMultipleInputClassifier` (Listing 6) extends
725 treatment response prediction to multiple data modalities. Its key components include:

- 726 • encoders: A collection of modality-specific encoder networks, organized as either a
727 `ModuleList` or `ModuleDict`
- 728 • classifier: A module that combines encoded representations to make predictions
- 729 • loss: The training objective function
- 730 • normalize: A flag indicating whether to normalize encoded representations before classifi-
731 cation

```

732 1 class MultiModalMultipleInputClassifier(LitClassifier):
733 2     """A multi-modal classifier.
734 3
735 4     This model takes multiple modalities as input and outputs a
736 5     prediction.
737 6
738 7     Attributes:
739 8         encoders: The encoder models.
740 9         classifier: The classifier model.
74110         loss: The loss function.
74211     """
74312
74413     encoders: nn.ModuleList | nn.ModuleDict
74514     classifier: nn.Module
74615     loss: nn.Module
74716     normalize: bool

```

Listing 6: Multi-modal classifier implementation.

748 A.4.5 Experiment Configuration

749 The experiment configuration system in `scGeneScope` is organized hierarchically by data modality
750 and profile type. The top-level structure is as follows:

```

experiment/
├── imaging/
│   ├── multiprofile/
│   └── singleprofile/
├── multimodal/
│   ├── multiprofile/
│   └── singleprofile/
└── rnaseq/
    ├── multiprofile/
    └── singleprofile/

```

751 After installing the project in its virtual environment, each experiment configuration can be executed
752 using the `train.py` script with the appropriate configuration file path. For example:

```
753 1 # Run single-profile imaging experiment with ViT-H
754 2 train experiment=imaging/singleprofile/train_on_concat_imagenet_vit_h.
755   yaml
756 3
757 4 # Run multi-profile RNA-seq experiment with PCA embeddings
758 5 train experiment=rnaseq/multiprofile/train_avgpool_on_pca_n2000.yaml
759 6
760 7 # Run multi-modal experiment combining ScGPT and ViT-L
761 8 train experiment=multimodal/multiprofile/
762   train_avgpool_on_scgpt_with_concat_imagenet_vit_l.yaml
```

763 The script automatically handles data loading, model initialization, training, and evaluation based on
764 the configuration parameters specified in the YAML files. Additional command-line arguments can
765 be passed to override configuration values:

```
766 1 # Explore extra parameters to override in src/scgenescope/config
767 2 train experiment=<config_path>
768 3   # Override trainer parameters
769 4   trainer.max_epochs=100 trainer.accelerator=gpu trainer.devices=[0]
770 5   # Override model parameters
771 6   model.optimizer_factory.lr=1e-4
```

772 **Imaging Configurations:** The imaging modality configurations support both single-profile and
773 multi-profile analyses using various vision transformer architectures:

```
imaging/multiprofile/
├─ train_avgpool_on_concat_imagenet_vit_h.yaml
├─ train_avgpool_on_concat_imagenet_vit_l.yaml
├─ train_deepset_on_concat_imagenet_vit_h.yaml
├─ train_deepset_on_concat_imagenet_vit_l.yaml
├─ train_transformerpool_on_concat_imagenet_vit_h.yaml
├─ train_transformerpool_on_concat_imagenet_vit_l.yaml
└─ imaging/singleprofile/
   ├─ train_on_concat_imagenet_vit_h.yaml
   ├─ train_on_concat_imagenet_vit_l.yaml
   ├─ train_on_concat_resnet152.yaml
   └─ train_on_concat_resnet50.yaml
```

774 **Multi-modal Configurations:** The multi-modal configurations integrate RNA-seq and imaging
775 data, with specialized setups for different embedding approaches:

```
multimodal/multiprofile/
├─ train_avgpool_on_pca_n2000_with_concat_imagenet_vit_h.yaml
├─ train_avgpool_on_pca_n2000_with_concat_imagenet_vit_l.yaml
├─ train_avgpool_on_scgpt_with_concat_imagenet_vit_h.yaml
├─ train_avgpool_on_scgpt_with_concat_imagenet_vit_l.yaml
├─ train_avgpool_on_scvi_n200_with_concat_imagenet_vit_h.yaml
├─ train_avgpool_on_scvi_n200_with_concat_imagenet_vit_l.yaml
└─ multimodal/singleprofile/
   ├─ train_on_pca_n2000_with_concat_imagenet_vit_h.yaml
   ├─ train_on_pca_n2000_with_concat_imagenet_vit_l.yaml
   ├─ train_on_scgpt_with_concat_imagenet_vit_h.yaml
   ├─ train_on_scgpt_with_concat_imagenet_vit_l.yaml
   ├─ train_on_scvi_n200_with_concat_imagenet_vit_h.yaml
   └─ train_on_scvi_n200_with_concat_imagenet_vit_l.yaml
```

776 **RNA-seq Configurations:** The RNA-seq configurations support various embedding approaches
777 and pooling strategies:

```
rnaseq/multiprofile/
└─ train_avgpool_on_pca_n2000.yaml
```

- └─ train_avgpool_on_scgpt.yaml
- └─ train_avgpool_on_scvi_n200.yaml
- └─ train_deepset_on_pca_n2000.yaml
- └─ train_deepset_on_scgpt.yaml
- └─ train_deepset_on_scvi_n200.yaml
- └─ train_transformerpool_on_pca_n2000.yaml
- └─ train_transformerpool_on_scgpt.yaml
- └─ train_transformerpool_on_scvi_n200.yaml
- └─ rnaseq/singleprofile/
 - └─ train_on_UCE_4layer.yaml
 - └─ train_on_geneformer.yaml
 - └─ train_on_pca_n2000.yaml
 - └─ train_on_scgpt.yaml
 - └─ train_on_scvi_1.yaml
 - └─ train_on_scvi_2.yaml
 - └─ train_on_scvi_n200.yaml

778 **B scGeneScope Datasheet**

779 **B.1 Motivation**

780 **B.1.1 For what purpose was the dataset created?**

781 The scGeneScope dataset was created for the express purpose of developing and evaluating unimodal
782 and multimodal Cell Painting and scRNA-seq machine learning methods for cellular phenotyping,
783 treatment classification, and MoA discovery simulation in conditionally paired multimodal settings.

784 **B.1.2 Who created the dataset?**

785 The scGeneScope was created by the Institute of Computation within Altos Labs to support this
786 research.

787 **B.1.3 Who funded the creation of the dataset?**

788 Altos Labs funded the creation of the scGeneScope dataset.

789 **B.1.4 Any other comments?**

790 No.

791 **B.2 Composition**

792 **B.2.1 What do the instances represent?**

793 Individual instances in the dataset represent Cell Painting or scRNA-seq measurements of individual
794 U2-OS cells perturbed with 1 of 28 chemicals referenced in Table 2 and Section A.1.

795 **B.2.2 How many instances are there in total?**

796 There are 627,704 scRNA-seq gene expression profiles and 716,767 crops of single cell Cell Painting
797 images.

798 **B.2.3 Is the dataset exhaustive or a sample of a larger set?**

799 The dataset is the exhaustive set of single cell images and scRNA-seq profiles which passed the
800 preprocessing and QC thresholds as described in Section A.1.

801 **B.2.4 What data does each instance consist of?**

802 The dataset consists of two types of data instances: 1) images of single cells (uint8, .tiff format)
803 stained using the Cell Painting protocol, and 2) vectors of single cell gene expression counts from the
804 data generation procedure as described in Section A.1.

805 **B.2.5 Is there a label or target for each instance?**

806 Each instance has an associated treatment label, which is used for the treatment identification task.

807 **B.2.6 Is any information missing from individual instances?**

808 No information is withheld from the individual instances.

809 **B.2.7 Are relationships between instances made explicit?**

810 The relationships between the instances are made explicit by the treatment identification and the
811 sample and replicate identifiers, which associate instances from the same treatment conditions, wells,
812 well plates, and processing batch.

813 **B.2.8 Recommended data splits (train/val/test)?**

814 There are a number of different potential splitting procedures that could be applied to the scGeneScope
815 dataset. This paper prescribes one possible splitting procedure which we use for our benchmarking
816 evaluations as described in Section 3.4.

817 **B.2.9 Errors, noise, or redundancies?**

818 To the best of our knowledge, the data does not contain any errors or redundancies, and only contains
819 standard noise associated with scRNA-seq and Cell Painting measurement modalities.

820 **B.2.10 Is the dataset self-contained or dependent on external resources?**

821 The dataset is self contained within two h5ad files containing the scRNA-seq data (one for round one
822 and one for round two) and two folder systems storing the Cell Painting imaging data. The data is
823 hosted at <https://huggingface.co/datasets/altoslabs/scGeneScope>, with tutorials on how to access and
824 use the data at our github, which will be made public at time of the paper's final release.

825 **B.2.11 Confidential data?**

826 There is no confidential data in the scGeneScope dataset.

827 **B.2.12 Potentially offensive or sensitive content?**

828 There is no offensive or sensitive content in the scGeneScope dataset.

829 **B.2.13 Does the dataset relate to people?**

830 The scGeneScope dataset does not relate to people.

831 **B.2.14 Identification of sub-populations?**

832 The scGeneScope does not identify any sub-populations.

833 **B.2.15 Possibility of identifying individuals?**

834 The scGeneScope does not identify any individuals.

835 **B.2.16 Sensitive attributes present?**

836 There are no sensitive attributes in the scGeneScope dataset.

837 **B.2.17 Any other comments?**

838 No.

839 **B.3 Collection Process**

840 **B.3.1 How was the data acquired?**

841 The data was acquired as described in Section A.1.

842 **B.3.2 Mechanisms or procedures used?**

843 The mechanisms and procedures used to generate the dataset are described in Section A.1, including
844 cell culture, single-cell RNA sequencing, and Cell Painting.

845 **B.3.3 Sampling strategy (if any)?**

846 All samples which passed quality control where used.

847 **B.3.4 Who was involved and how were they compensated?**

848 Altos Labs scientists were involved and compensated through their employment at Altos Labs.

849 **B.3.5 Timeframe of data collection?**

850 The dataset was collected in two rounds, with one round collected over the course of three weeks in
851 early 2024 and another round collected over the course of a week in early 2025.

852 **B.3.6 Ethical review processes?**

853 There was no ethical review processes, as it was deemed unnecessary for the contents of this dataset.

854 **B.3.7 If people are involved:**

855 No people were involved beyond the scientists of Altos Labs who generated the dataset.

856 **B.3.8 Any other comments?**

857 No.

858 **B.4 Pre-processing / Cleaning / Labeling**

859 **B.4.1 Was any pre-processing done?**

860 The Cell Painting data was processed according to the procedure described in Section A.1.4, and the
861 scRNA-seq data was preprocessed according to the procedure described in Section A.1.4.

862 **B.4.2 Is raw data saved in addition to processed data?**

863 At the time of paper submission, the raw data is saved but not yet released alongside the preprocessed
864 data for technical and storage related reasons. We release both the imaging and scRNAseq data in
865 preprocessed forms commonly used by the field, and plan to make the full, raw data available at the
866 time of the final paper release.

867 **B.4.3 Is the preprocessing software available?**

868 All of the preprocessing software and methods we used for both imaging and scRNAseq data are
869 publicly available and described in Section A.1.4.

870 **B.4.4 Any other comments?**

871 Regarding the raw data, due to the sheer volume of the raw data compared to the preprocessed data,
872 it was technically challenging to find a hosting service and get the data in place by the time of the
873 NeurIPS 2025 submission. We are firmly of the belief that the data is most valuable to our community
874 when shared in both the easily accessible preprocessed form (as it is currently) and also in the raw
875 form which will allow the community to explore different data processing methods or use cases. As
876 such, we are committed to sharing the raw data as well, and are working on technical solutions to
877 make this feasible.

878 **B.5 Uses**

879 **B.5.1 Has the dataset been used for any tasks already?**

880 This is the first task that this dataset has been used for.

881 **B.5.2 Repository of papers/systems using the dataset?**

882 This is the first paper that has used this dataset.

883 **B.5.3 Other potential tasks?**

884 This dataset has potential for a variety of additional tasks and use cases related to unimodal and
885 multimodal data processing.

886 **B.5.4 Factors that might impact future uses?**

887 **B.5.5 Tasks for which the dataset *should not* be used?**

888 The scGeneScope dataset should not be used for any tasks that involve making treatment classification
889 predictions where the two replicates from the second round of data generation are split between train
890 and test sets. This is because there is a confounding batch effect in round two of data generation,
891 where in groups of seven treatments for both replicates are processed in the same batch during this
892 generation procedure. This leads to a confounding effect, where models can use the batch signal to
893 narrow their predictions to treatments within the same batch, and artificially increase their scores.
894 Round one of generation does not have this effect. See Figure 3 and Section A.1.1. For this reason,
895 it is recommended to only use the generation round one replicates in either the test set (as we have
896 done) or the train set, but not both.

897 **B.5.6 Any other comments?**

898 No.

899 **B.6 Distribution**

900 **B.6.1 Will the dataset be distributed to third parties?**

901 The scGeneScope dataset is released for non-commercial use under a CC BY-NC 4.0 license at
902 <https://huggingface.co/datasets/altoslabs/scGeneScope>.

903 **B.6.2 Distribution method (tarball, API, GitHub, DOI)?**

904 The scGeneScope dataset is hosted at <https://huggingface.co/datasets/altoslabs/scGeneScope>,
905 with download and usage instructions in our code. The code has been shared with
906 reviewers, and will be made publicly available for download under the CC BY-NC 4.0 license at the
907 time of final paper release.

908 **B.6.3 When will the dataset be distributed?**

909 The scGeneScope dataset was made public at the time of the NeurIPS 2025 submission, May 15th,
910 2025.

911 **B.6.4 License or terms of use?**

912 The dataset is licensed under a CC BY-NC 4.0 license for non commercial use.

913 **B.6.5 Third-party IP or other restrictions?**

914 The scGeneScope has no third-party IP or other restrictions.

915 **B.6.6 Export controls or regulatory restrictions?**

916 The scGeneScope has no export controls or regulatory restrictions.

917 **B.6.7 Any other comments?**

918 No.

919 **B.7 Maintenance**

920 **B.7.1 Who supports / hosts / maintains the dataset?**

921 Altos Labs supports and maintains the dataset, and HuggingFace.co hosts the dataset.

922 **B.7.2 Contact information for the owner/curator/manager?**

923 Please contact the primary contact of this paper, which will be included at the time of the paper's
924 final release.

925 **B.7.3 Erratum available?**

926 **B.7.4 Will the dataset be updated?**

927 Yes, the dataset will be updated as we make the full raw data available.

928 **B.7.5 Retention limits for data relating to people?**

929 The data has no retention limits or relation to people.

930 **B.7.6 Will older versions remain available?**

931 The current version will remain available.

932 **B.7.7 Mechanism for external contributions / extensions?**

933 The dataset could in theory be extended by incorporation into other composite datasets like CELLx-
934 GENE though we would caution against including it into large atlas based training sets, as we see the
935 main utility of this dataset as a benchmarking dataset rather than a pretraining dataset.

936 **B.7.8 Any other comments?**

937 No.

References

- [1] Mark-Anthony Bray, Shantanu Singh, Han Han, Chadwick T Davis, Blake Borgeson, Cathy Hartland, Maria Kost-Alimova, Sigrun M Gustafsdottir, Christopher C Gibson, and Anne E Carpenter. Cell painting, a high-content image-based assay for morphological profiling using multiplexed fluorescent dyes. *Nature protocols*, 11(9):1757–1774, 2016.
- [2] Aravind Subramanian, Rajiv Narayan, Steven M. Corsello, David D. Peck, Ted Natoli, Xiaodong Lu, Joshua Gould, John Davis, Andrew Tubelli, Jacob Asiedu, David Lahr, Jodi Hirschman, Zihan Liu, Melanie Donahue, Bina Julian, Mariya Khan, David Wadden, Ian Smith, Daniel Lam, Arthur Liberzon, Courtney Toder, Mukta Bagul, Marek Orzechowski, Oana Enache, Federica Piccioni, Sarah Johnson, Nicholas Lyons, Alice Berger, Jesse B. Boehm, Stuart L. Schreiber, Justin Lamb, and Todd R. Golub. A next generation connectivity map: L1000 platform and the first 1,000,000 profiles. *Cell*, 171(6):1437–1452.e17, 2017.
- [3] Fuchou Tang, Catalin Barbacioru, Yangzhou Wang, Ellen Nordman, Clarence Lee, Nanlan Xu, Xiaohui Wang, John Bodeau, Brian B. Tuch, Asim Siddiqui, Kaiqin Lao, and M. Azim Surani. mrna-seq whole-transcriptome analysis of a single cell. *Nature Methods*, 6(5):377–382, 2009.
- [4] Gregory P. Way, Ted Natoli, Adeniyi Adeboye, Lev Litichevskiy, Andrew Yang, Xiaodong Lu, Juan C. Caicedo, Beth A. Cimini, Kyle Karhohs, David J. Logan, Mohammad H. Rohban, Maria Kost-Alimova, Kate Hartland, Michael Bornholdt, Srinivas N. Chandrasekaran, Marzieh Haghighi, Erin Weisbart, Shantanu Singh, Aravind Subramanian, and Anne E. Carpenter. Morphology and gene expression profiling provide complementary information for mapping cell state. *Cell Systems*, 13(11):911–923.e9, 2022.
- [5] Marzieh Haghighi, Juan C Caicedo, Beth A Cimini, Anne E Carpenter, and Shantanu Singh. High-dimensional gene expression and morphology profiles of cells across 28,000 genetic and chemical perturbations. *Nature methods*, 19(12):1550–1557, 2022.
- [6] Romain Lopez, Jeffrey Regier, Michael B. Cole, Michael I. Jordan, and Nir Yosef. Deep generative modeling for single-cell transcriptomics. *Nature Methods*, 15(12):1053–1058, 2018.
- [7] Srinivas Niranj Chandrasekaran, Beth A. Cimini, Amy Goodale, Lisa Miller, Maria Kost-Alimova, Nasim Jamali, John G. Doench, Briana Fritchman, Adam Skepner, Michelle Melanson, Alexandr A. Kalinin, John Arevalo, Marzieh Haghighi, Juan C. Caicedo, Daniel Kuhn, Desiree Hernandez, James Berstler, Hamdah Shafqat-Abbasi, David E. Root, Susanne E. Swalley, Sakshi Garg, Shantanu Singh, and Anne E. Carpenter. Three million images and morphological profiles of cells treated with matched chemical and genetic perturbations. *Nature Methods*, 21:1114–1121, 2024.
- [8] Recursion Pharmaceuticals. RxRx3: Phenomics Map of Biology Dataset. <https://www.rxxr.ai/rxxr3>, 2024. Accessed 1 May 2025.
- [9] Nikita Moshkov, Michael Bornholdt, Santiago Benoit, Matthew Smith, Claire McQuin, Allen Goodman, Rebecca A. Senft, Yu Han, Mehrtash Babadi, Peter Horvath, Beth A. Cimini, Anne E. Carpenter, Shantanu Singh, and Juan C. Caicedo. Learning representations for image-based profiling of perturbations. *Nature Communications*, 15:1594, 2024.
- [10] Shibli Abdulla, Brian D. Aeversmann, Pedro Assis, Seve Badajoz, Sidney M. Bell, Emanuele Bezzi, Batuhan Cakir, Jim Chaffer, and Ambrose Carr. Cz cell×gene discover: A single-cell data platform for scalable exploration, analysis and modeling of aggregated data. *bioRxiv*, 2023.
- [11] Aviv Regev, Sarah A. Teichmann, Eric S. Lander, Ido Amit, Christophe Benoist, Ewan Birney, Bernd Bodenmiller, Peter Campbell, Piero Carninci, Menna Clatworthy, Hans Clevers, Bart Deplancke, Ian Dunham, James Eberwine, Roland Eils, Wolfgang Enard, Andrew Farmer, Lars Fugger, Berthold Göttgens, Nir Hacohen, Muzlifah Haniffa, Martin Hemberg, Seung Kim, Paul Klenerman, Arnold Kriegstein, Ed Lein, Sten Linnarsson, Emma Lundberg, Joakim Lundberg, Partha Majumder, John C. Marioni, Miriam Merad, Musa Mhlanga, Martijn Nawijn, Mihai Netea, Garry Nolan, Dana Pe’er, Anthony Phillipakis, Chris P. Ponting, Stephen Quake, Wolf Reik, Orit Rozenblatt-Rosen, Joshua Sanes, Rahul Satija, Ton N. Schumacher, Alex Shalek, Ehud Shapiro, Padmanee Sharma, Jay W. Shin, Oliver Stegle, Michael Stratton, Michael J. T.

- 989 Stubbington, Fabian J. Theis, Mathias Uhlén, Alexander van Oudenaarden, Allon Wagner,
990 Fiona Watt, Jonathan Weissman, Barbara Wold, Ramnik Xavier, Nir Yosef, and Human Cell
991 Atlas Meeting Participants. The human cell atlas. *eLife*, 6:e27041, December 2017.
- 992 [12] Joshua A. Harrill, Logan J. Everett, Derik E. Haggard, Thomas Sheffield, Joseph L. Bundy,
993 Clinton M. Willis, Russell S. Thomas, Imran Shah, and Richard S. Judson. High-throughput tran-
994 scriptomics platform for screening environmental chemicals. *Toxicological Sciences*, 181(1):68–
995 89, 2021.
- 996 [13] Jo Nyffeler, Clinton Willis, Felix R. Harris, M. J. Foster, Bryant Chambers, Megan Culbreth,
997 Richard E. Brockway, Sarah Davidson–Fritz, Daniel Dawson, Imran Shah, Katie Paul Friedman,
998 Dan Chang, Logan J. Everett, John F. Wambaugh, Grace Patlewicz, and Joshua A. Harrill. Ap-
999 plication of cell painting for chemical hazard evaluation in support of screening-level chemical
1000 assessments. *Toxicology and Applied Pharmacology*, 468:116513, 2023.
- 1001 [14] Christa Haase, Karin Gustafsson, Shenglin Mei, Shu-Chi Yeh, Dmitry Richter, Jelena Milosevic,
1002 Raphaël Turcotte, Peter V. Kharchenko, David B. Sykes, David T. Scadden, and Charles P.
1003 Lin. Image-seq: Spatially resolved single-cell sequencing guided by in situ imaging. *Nature*
1004 *Methods*, 19:1622–1633, 2022.
- 1005 [15] Olivia Padovan-Merhar, Gautham P. Nair, Andrew G. Biaesch, Andreas Mayer, Steven Scarfone,
1006 Shawn W. Foley, Angela R. Wu, L. Stirling Churchman, Abhyudai Singh, and Arjun Raj. Single
1007 mammalian cells compensate for differences in cellular volume and dna copy number through
1008 independent global transcriptional mechanisms. *Molecular Cell*, 58(2):339–352, 2015.
- 1009 [16] Haotian Cui, Chloe Wang, Hassaan Maan, Kuan Pang, Fengning Luo, Nan Duan, and Bo Wang.
1010 scgpt: Toward building a foundation model for single-cell multi-omics using generative ai.
1011 *Nature Methods*, 21:1470–1480, 2024.
- 1012 [17] Christina V. Theodoris, Ling Xiao, Anant Chopra, Mark D. Chaffin, Zeina R. Al Sayed,
1013 Matthew C. Hill, Helene Mantineo, Elizabeth M. Brydon, Zexian Zeng, X. Shirley Liu,
1014 and Patrick T. Ellinor. Transfer learning enables predictions in network biology. *Nature*,
1015 618(7965):616–624, 2023.
- 1016 [18] Yanay Rosen, Yusuf Roohani, Ayush Agarwal, Leon Samotorcan, Stephen R. Quake, and Jure
1017 Leskovec. Universal cell embeddings: A foundation model for cell biology. *bioRxiv*, 2023.
- 1018 [19] Rebecca Boiarsky, Nalini M. Singh, Alejandro Buendia, Ava P. Amini, Gad Getz, and David
1019 Sontag. Deeper evaluation of a single-cell foundation model. *Nature Machine Intelligence*,
1020 6(12):1443–1446, 2024. Matters Arising; shows that logistic regression can outperform the
1021 zero-shot scBERT RNA-seq foundation model.
- 1022 [20] Kasia Z. Kedzierska, Lorin Crawford, Ava P. Amini, and Alex X. Lu. Zero-shot evaluation
1023 reveals limitations of single-cell foundation models. *Genome Biology*, 26(1):101, 2025.
- 1024 [21] Anne E. Carpenter, Thouis R. Jones, Michael R. Lamprecht, Colin Clarke, In Han Kang, Ola
1025 Friman, David A. Guertin, Joo Han Chang, Robert A. Lindquist, Jason Moffat, Polina Golland,
1026 and David M. Sabatini. Cellprofiler: image analysis software for identifying and quantifying
1027 cell phenotypes. *Genome Biology*, 7:R100, 2006.
- 1028 [22] Nick Pawlowski, C. Caicedo Juan Shantanu Singh, E. Carpenter Anne and Amos Storkey.
1029 Automating morphological profiling with generic deep convolutional networks. *bioRxiv*, 2016.
1030 Posted 2 November 2016.
- 1031 [23] D. Michael Ando, Y. McLean Cory and Marc Berndl. Improving phenotypic measurements in
1032 high-content imaging screens. *bioRxiv*, 2017. Posted 10 July 2017.
- 1033 [24] Oren Kraus, Kian Kenyon-Dean, Saber Saberian, Maryam Fallah, Peter McLean, Jess Le-
1034 ung, Vasudev Sharma, Ayla Khan, Jia Balakrishnan, Safiye Celik, Dominique Beaini, Maciej
1035 Sypetkowski, Chi Vicky Cheng, Kristen Morse, Maureen Makes, Ben Mabey, and Berton
1036 Earnshaw. Masked autoencoders for microscopy are scalable learners of cellular biology. In
1037 2024 *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages
1038 11757–11768, 2024.

- [25] Heming Yao, Phil Hanslovsky, Jan-Christian Huetter, Burkhard Hoeckendorf, and David Richmond. Weakly supervised set-consistency learning improves morphological profiling of single-cell images. In *2024 IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops (CVPRW)*, pages 6978–6987, 2024.
- [26] J. Pontén and E. Saksela. Two established *In Vitro* cell lines from human mesenchymal tumours. *International Journal of Cancer*, 2(5):434–447, 1967.
- [27] Louise Heinrich, Karl Kumbier, Li Li, Steven M. Altschuler, and Lani F. Wu. Selection of optimal cell lines for high-content phenotypic screening. *ACS Chemical Biology*, 18(4):679–685, 2023.
- [28] Sebastian Nijman. Functional genomics to uncover drug mechanism of action. *Nature chemical biology*, 11(12):942–948, 2015.
- [29] Elisabet Gregori-Puigjané, Vincent Setola, Jérôme Hert, Brenda A Crews, John J Irwin, Eugen Lounkine, Lawrence Marnett, Bryan L Roth, and Brian K Shoichet. Identifying mechanism-of-action targets for drugs and probes. *Proceedings of the National Academy of Sciences*, 109(28):11178–11183, 2012.
- [30] James R Heath, Antoni Ribas, and Paul S Mischel. Single-cell analysis tools for drug discovery and development. *Nature reviews Drug discovery*, 15(3):204–216, 2016.
- [31] Tianyu Liu, Edward De Brouwer, Tony Kuo, Nathaniel Diamant, Alsu Missarova, Hanchen Wang, Minsheng Hao, Tommaso Biancalani, Hector Corrada Bravo, Gabriele Scalia, Aviv Regev, and Graham Heimberg. Learning multi-cellular representations of single-cell transcriptomics data enables characterization of patient-level disease states. *bioRxiv*, 2024.
- [32] Robert van Dijk, John Arevalo, Mehrtash Babadi, Anne E. Carpenter, and Shantanu Singh. Capturing cell heterogeneity in representations of cell populations for image-based profiling using contrastive learning. *PLOS Computational Biology*, 20(11):1–23, 11 2024.
- [33] Jelena Čuklina, Patrick GA Pedrioli, and Ruedi Aebersold. Review of batch effects prevention, diagnostics, and correction approaches. In *Mass spectrometry data analysis in proteomics*, pages 373–387. Springer, 2019.
- [34] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. ImageNet: A large-scale hierarchical image database. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 248–255, 2009.
- [35] PyTorch Core Team. torchvision: Computer vision utilities for PyTorch. <https://pytorch.org/vision/>, 2016.
- [36] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 770–778, 2016.
- [37] Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, Jakob Uszkoreit, and Neil Houlsby. An image is worth 16×16 words: Transformers for image recognition at scale. In *International Conference on Learning Representations (ICLR)*, 2021. arXiv:2010.11929.
- [38] Stanley Bryan Z Hua, Alex X Lu, and Alan M Moses. Cytoimagenet: A large-scale pretraining dataset for bioimage transfer learning. *arXiv preprint arXiv:2111.11646*, 2021.
- [39] Chan Zuckerberg CELL×GENE Team. scVI-1 pretrained model on CELL×GENE Census (release 2023-05-15). <https://cellxgene.cziscience.com/census-models>, 2023.
- [40] Chan Zuckerberg CELL×GENE Team. scVI-2 pretrained model on CELL×GENE Census (release 2024-02-12). <https://cellxgene.cziscience.com/census-models>, 2024.

- 1085 [41] Manzil Zaheer, Satwik Kottur, Siamak Ravanbakhsh, Barnabas Poczos, Russ R Salakhutdinov,
1086 and Alexander J Smola. Deep sets. In I. Guyon, U. Von Luxburg, S. Bengio, H. Wallach,
1087 R. Fergus, S. Vishwanathan, and R. Garnett, editors, *Advances in Neural Information Processing*
1088 *Systems*, volume 30. Curran Associates, Inc., 2017.
- 1089 [42] Juho Lee, Yoonho Lee, Jungtaek Kim, Adam Kosiorek, Seungjin Choi, and Yee Whye Teh.
1090 Set transformer: A framework for attention-based permutation-invariant neural networks. In
1091 Kamalika Chaudhuri and Ruslan Salakhutdinov, editors, *Proceedings of the 36th International*
1092 *Conference on Machine Learning*, volume 97 of *Proceedings of Machine Learning Research*,
1093 pages 3744–3753. PMLR, 09–15 Jun 2019.
- 1094 [43] Ricard Argelaguet, Anna SE Cuomo, Oliver Stegle, and John C Marioni. Computational
1095 principles and challenges in single-cell data integration. *Nature biotechnology*, 39(10):1202–
1096 1215, 2021.
- 1097 [44] Jesse Zhang, Airol A. Ubas, Richard de Borja, Valentine Svensson, Nicole Thomas, Neha
1098 Thakar, Ian Lai, Aidan Winters, Umair Khan, Matthew G. Jones, Vuong Tran, Joseph Pangallo,
1099 Efthymia Papalexi, Ajay Sapre, Hoai Nguyen, Oliver Sanderson, Maria Nigos, Olivia Kaplan,
1100 Sarah Schroeder, Bryan Hariadi, and Simone Marrujo. Tahoe-100m: A giga-scale single-cell
1101 perturbation atlas for context-dependent gene function and cellular modeling. *bioRxiv*, 2025.
- 1102 [45] David S Fischer, Martin A Villanueva, Peter S Winter, and Alex K Shalek. Adapting systems
1103 biology to address the complexity of human disease in the single-cell era. *Nature Reviews*
1104 *Genetics*, pages 1–18, 2025.
- 1105 [46] Mark A. Bray, Saga M. Gustafsdottir, Mohammad H. Rohban, Shantanu Singh, Vebjorn Ljosa,
1106 Kelly L. Sokolnicki, Joshua A. Bittker, Nicole E. Bodycombe, Vlado Dancík, Timothy P.
1107 Hasaka, Chantal S. Hon, Maria M. Kemp, Kan Li, Dilanthi Walpita, Mathias J. Wawer, Todd R.
1108 Golub, Stuart L. Schreiber, Paul A. Clemons, Alykhan F. Shamji, and Anne E. Carpenter. A
1109 dataset of images and morphological profiles of 30 000 small-molecule treatments using the cell
1110 painting assay. *GigaScience*, 6(12):1–5, Dec 2017.
- 1111 [47] Zaber Technologies Inc. Microscope autofocus with python and opencv. [https://github.](https://github.com/zabertech/zaber-examples/tree/main/examples/microscope_autofocus)
1112 [com/zabertech/zaber-examples/tree/main/examples/microscope_autofocus](https://github.com/zabertech/zaber-examples/tree/main/examples/microscope_autofocus),
1113 2024.
- 1114 [48] Carsen Stringer, Michalis Michaelos, and Marius Pachitariu. Cellpose: a generalist algorithm
1115 for cellular segmentation. *Nature Methods*, 18(1):100–106, 2021.
- 1116 [49] Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan,
1117 Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, Alban Desmaison, Andreas
1118 Köpf, Edward Z. Yang, Zach DeVito, Martin Raison, Alykhan Tejani, Sasank Chilamkurthy,
1119 Benoit Steiner, Lu Fang, Junjie Bai, and Soumith Chintala. Pytorch: An imperative style,
1120 high-performance deep learning library. *Advances in Neural Information Processing Systems*,
1121 abs/1912.01703, 2019.
- 1122 [50] William Falcon and The PyTorch Lightning team. PyTorch Lightning, March 2019.
- 1123 [51] Omry Yadan. Hydra - a framework for elegantly configuring complex applications. Github,
1124 2019.
- 1125 [52] F Alexander Wolf, Philipp Angerer, and Fabian J Theis. Scanpy: large-scale single-cell gene
1126 expression data analysis. *Genome biology*, 19:1–5, 2018.

1127 **NeurIPS Paper Checklist**

1128 **1. Claims**

1129 Question: Do the main claims made in the abstract and introduction accurately reflect the
1130 paper's contributions and scope?

1131 Answer: [\[Yes\]](#)

1132 Justification: We only make claims in our abstract and introduction that are directly supported
1133 by our dataset and benchmarking observations.

1134 Guidelines:

- 1135 • The answer NA means that the abstract and introduction do not include the claims
1136 made in the paper.
- 1137 • The abstract and/or introduction should clearly state the claims made, including the
1138 contributions made in the paper and important assumptions and limitations. A No or
1139 NA answer to this question will not be perceived well by the reviewers.
- 1140 • The claims made should match theoretical and experimental results, and reflect how
1141 much the results can be expected to generalize to other settings.
- 1142 • It is fine to include aspirational goals as motivation as long as it is clear that these goals
1143 are not attained by the paper.

1144 **2. Limitations**

1145 Question: Does the paper discuss the limitations of the work performed by the authors?

1146 Answer: [\[Yes\]](#)

1147 Justification: In the Discussion section, we clearly state the limitations of both our dataset
1148 and our benchmarking efforts, including which baselines we believe are missing.

1149 Guidelines:

- 1150 • The answer NA means that the paper has no limitation while the answer No means that
1151 the paper has limitations, but those are not discussed in the paper.
- 1152 • The authors are encouraged to create a separate "Limitations" section in their paper.
- 1153 • The paper should point out any strong assumptions and how robust the results are to
1154 violations of these assumptions (e.g., independence assumptions, noiseless settings,
1155 model well-specification, asymptotic approximations only holding locally). The authors
1156 should reflect on how these assumptions might be violated in practice and what the
1157 implications would be.
- 1158 • The authors should reflect on the scope of the claims made, e.g., if the approach was
1159 only tested on a few datasets or with a few runs. In general, empirical results often
1160 depend on implicit assumptions, which should be articulated.
- 1161 • The authors should reflect on the factors that influence the performance of the approach.
1162 For example, a facial recognition algorithm may perform poorly when image resolution
1163 is low or images are taken in low lighting. Or a speech-to-text system might not be
1164 used reliably to provide closed captions for online lectures because it fails to handle
1165 technical jargon.
- 1166 • The authors should discuss the computational efficiency of the proposed algorithms
1167 and how they scale with dataset size.
- 1168 • If applicable, the authors should discuss possible limitations of their approach to
1169 address problems of privacy and fairness.
- 1170 • While the authors might fear that complete honesty about limitations might be used by
1171 reviewers as grounds for rejection, a worse outcome might be that reviewers discover
1172 limitations that aren't acknowledged in the paper. The authors should use their best
1173 judgment and recognize that individual actions in favor of transparency play an impor-
1174 tant role in developing norms that preserve the integrity of the community. Reviewers
1175 will be specifically instructed to not penalize honesty concerning limitations.

1176 **3. Theory assumptions and proofs**

1177 Question: For each theoretical result, does the paper provide the full set of assumptions and
1178 a complete (and correct) proof?

Answer: [NA]

Justification: We do not include any theoretical work in our paper.

Guidelines:

- The answer NA means that the paper does not include theoretical results.
- All the theorems, formulas, and proofs in the paper should be numbered and cross-referenced.
- All assumptions should be clearly stated or referenced in the statement of any theorems.
- The proofs can either appear in the main paper or the supplemental material, but if they appear in the supplemental material, the authors are encouraged to provide a short proof sketch to provide intuition.
- Inversely, any informal proof provided in the core of the paper should be complemented by formal proofs provided in appendix or supplemental material.
- Theorems and Lemmas that the proof relies upon should be properly referenced.

4. Experimental result reproducibility

Question: Does the paper fully disclose all the information needed to reproduce the main experimental results of the paper to the extent that it affects the main claims and/or conclusions of the paper (regardless of whether the code and data are provided or not)?

Answer: [Yes]

Justification: We include clear details and instructions in our paper about what operations we performed to obtain our code, and we also include our code and data as supplementary material demonstrating how to reproduce our results.

Guidelines:

- The answer NA means that the paper does not include experiments.
- If the paper includes experiments, a No answer to this question will not be perceived well by the reviewers: Making the paper reproducible is important, regardless of whether the code and data are provided or not.
- If the contribution is a dataset and/or model, the authors should describe the steps taken to make their results reproducible or verifiable.
- Depending on the contribution, reproducibility can be accomplished in various ways. For example, if the contribution is a novel architecture, describing the architecture fully might suffice, or if the contribution is a specific model and empirical evaluation, it may be necessary to either make it possible for others to replicate the model with the same dataset, or provide access to the model. In general, releasing code and data is often one good way to accomplish this, but reproducibility can also be provided via detailed instructions for how to replicate the results, access to a hosted model (e.g., in the case of a large language model), releasing of a model checkpoint, or other means that are appropriate to the research performed.
- While NeurIPS does not require releasing code, the conference does require all submissions to provide some reasonable avenue for reproducibility, which may depend on the nature of the contribution. For example
 - (a) If the contribution is primarily a new algorithm, the paper should make it clear how to reproduce that algorithm.
 - (b) If the contribution is primarily a new model architecture, the paper should describe the architecture clearly and fully.
 - (c) If the contribution is a new model (e.g., a large language model), then there should either be a way to access this model for reproducing the results or a way to reproduce the model (e.g., with an open-source dataset or instructions for how to construct the dataset).
 - (d) We recognize that reproducibility may be tricky in some cases, in which case authors are welcome to describe the particular way they provide for reproducibility. In the case of closed-source models, it may be that access to the model is limited in some way (e.g., to registered users), but it should be possible for other researchers to have some path to reproducing or verifying the results.

5. Open access to data and code

1233 Question: Does the paper provide open access to the data and code, with sufficient instruc-
1234 tions to faithfully reproduce the main experimental results, as described in supplemental
1235 material?

1236 Answer: [Yes]

1237 Justification: We release our dataset first to reviewers in during the review period via
1238 HuggingFace with a private access token, and we will release our data in full to the public if
1239 and when our paper is accepted. We release our code via github, with sufficient instructions
1240 to reproduce our main results.

1241 Guidelines:

- 1242 • The answer NA means that paper does not include experiments requiring code.
- 1243 • Please see the NeurIPS code and data submission guidelines (<https://nips.cc/public/guides/CodeSubmissionPolicy>) for more details.
- 1244 • While we encourage the release of code and data, we understand that this might not be
1245 possible, so “No” is an acceptable answer. Papers cannot be rejected simply for not
1246 including code, unless this is central to the contribution (e.g., for a new open-source
1247 benchmark).
- 1248 • The instructions should contain the exact command and environment needed to run to
1249 reproduce the results. See the NeurIPS code and data submission guidelines (<https://nips.cc/public/guides/CodeSubmissionPolicy>) for more details.
- 1250 • The authors should provide instructions on data access and preparation, including how
1251 to access the raw data, preprocessed data, intermediate data, and generated data, etc.
- 1252 • The authors should provide scripts to reproduce all experimental results for the new
1253 proposed method and baselines. If only a subset of experiments are reproducible, they
1254 should state which ones are omitted from the script and why.
- 1255 • At submission time, to preserve anonymity, the authors should release anonymized
1256 versions (if applicable).
- 1257 • Providing as much information as possible in supplemental material (appended to the
1258 paper) is recommended, but including URLs to data and code is permitted.

1261 6. Experimental setting/details

1262 Question: Does the paper specify all the training and test details (e.g., data splits, hyper-
1263 parameters, how they were chosen, type of optimizer, etc.) necessary to understand the
1264 results?

1265 Answer: [Yes]

1266 Justification: We detail our training and testing splits and rationale in detail in the main paper,
1267 and include the hyperparameters and optimization choices in the supplementary material
1268 and in our code.

1269 Guidelines:

- 1270 • The answer NA means that the paper does not include experiments.
- 1271 • The experimental setting should be presented in the core of the paper to a level of detail
1272 that is necessary to appreciate the results and make sense of them.
- 1273 • The full details can be provided either with the code, in appendix, or as supplemental
1274 material.

1275 7. Experiment statistical significance

1276 Question: Does the paper report error bars suitably and correctly defined or other appropriate
1277 information about the statistical significance of the experiments?

1278 Answer: [Yes]

1279 Justification: All of our results are reported with standard error over 5 training seeds.

1280 Guidelines:

- 1281 • The answer NA means that the paper does not include experiments.
- 1282 • The authors should answer "Yes" if the results are accompanied by error bars, confi-
1283 dence intervals, or statistical significance tests, at least for the experiments that support
1284 the main claims of the paper.

- The factors of variability that the error bars are capturing should be clearly stated (for example, train/test split, initialization, random drawing of some parameter, or overall run with given experimental conditions).
- The method for calculating the error bars should be explained (closed form formula, call to a library function, bootstrap, etc.)
- The assumptions made should be given (e.g., Normally distributed errors).
- It should be clear whether the error bar is the standard deviation or the standard error of the mean.
- It is OK to report 1-sigma error bars, but one should state it. The authors should preferably report a 2-sigma error bar than state that they have a 96% CI, if the hypothesis of Normality of errors is not verified.
- For asymmetric distributions, the authors should be careful not to show in tables or figures symmetric error bars that would yield results that are out of range (e.g. negative error rates).
- If error bars are reported in tables or plots, The authors should explain in the text how they were calculated and reference the corresponding figures or tables in the text.

8. Experiments compute resources

Question: For each experiment, does the paper provide sufficient information on the computer resources (type of compute workers, memory, time of execution) needed to reproduce the experiments?

Answer: [Yes]

Justification: Yes, in our supplementary material we include details of the compute resources used and estimates of the memory and time of execution.

Guidelines:

- The answer NA means that the paper does not include experiments.
- The paper should indicate the type of compute workers CPU or GPU, internal cluster, or cloud provider, including relevant memory and storage.
- The paper should provide the amount of compute required for each of the individual experimental runs as well as estimate the total compute.
- The paper should disclose whether the full research project required more compute than the experiments reported in the paper (e.g., preliminary or failed experiments that didn't make it into the paper).

9. Code of ethics

Question: Does the research conducted in the paper conform, in every respect, with the NeurIPS Code of Ethics <https://neurips.cc/public/EthicsGuidelines>?

Answer: [Yes]

Justification: Our paper contains no violations to the NeurIPS Code of Ethics.

Guidelines:

- The answer NA means that the authors have not reviewed the NeurIPS Code of Ethics.
- If the authors answer No, they should explain the special circumstances that require a deviation from the Code of Ethics.
- The authors should make sure to preserve anonymity (e.g., if there is a special consideration due to laws or regulations in their jurisdiction).

10. Broader impacts

Question: Does the paper discuss both potential positive societal impacts and negative societal impacts of the work performed?

Answer: [Yes]

Justification: Our paper includes discussion of the potential positive impacts including the development of better biological foundation models. We do not recognize any potential for negative societal impacts of our dataset or models.

Guidelines:

- The answer NA means that there is no societal impact of the work performed.
- If the authors answer NA or No, they should explain why their work has no societal impact or why the paper does not address societal impact.
- Examples of negative societal impacts include potential malicious or unintended uses (e.g., disinformation, generating fake profiles, surveillance), fairness considerations (e.g., deployment of technologies that could make decisions that unfairly impact specific groups), privacy considerations, and security considerations.
- The conference expects that many papers will be foundational research and not tied to particular applications, let alone deployments. However, if there is a direct path to any negative applications, the authors should point it out. For example, it is legitimate to point out that an improvement in the quality of generative models could be used to generate deepfakes for disinformation. On the other hand, it is not needed to point out that a generic algorithm for optimizing neural networks could enable people to train models that generate Deepfakes faster.
- The authors should consider possible harms that could arise when the technology is being used as intended and functioning correctly, harms that could arise when the technology is being used as intended but gives incorrect results, and harms following from (intentional or unintentional) misuse of the technology.
- If there are negative societal impacts, the authors could also discuss possible mitigation strategies (e.g., gated release of models, providing defenses in addition to attacks, mechanisms for monitoring misuse, mechanisms to monitor how a system learns from feedback over time, improving the efficiency and accessibility of ML).

11. Safeguards

Question: Does the paper describe safeguards that have been put in place for responsible release of data or models that have a high risk for misuse (e.g., pretrained language models, image generators, or scraped datasets)?

Answer: [NA]

Justification: We do not believe that our dataset, benchmarks, or results have any significant potential for misuse.

Guidelines:

- The answer NA means that the paper poses no such risks.
- Released models that have a high risk for misuse or dual-use should be released with necessary safeguards to allow for controlled use of the model, for example by requiring that users adhere to usage guidelines or restrictions to access the model or implementing safety filters.
- Datasets that have been scraped from the Internet could pose safety risks. The authors should describe how they avoided releasing unsafe images.
- We recognize that providing effective safeguards is challenging, and many papers do not require this, but we encourage authors to take this into account and make a best faith effort.

12. Licenses for existing assets

Question: Are the creators or original owners of assets (e.g., code, data, models), used in the paper, properly credited and are the license and terms of use explicitly mentioned and properly respected?

Answer: [Yes]

Justification: We are the creators and original owners of all novel data assets disclosed in this paper.

Guidelines:

- The answer NA means that the paper does not use existing assets.
- The authors should cite the original paper that produced the code package or dataset.
- The authors should state which version of the asset is used and, if possible, include a URL.
- The name of the license (e.g., CC-BY 4.0) should be included for each asset.

- For scraped data from a particular source (e.g., website), the copyright and terms of service of that source should be provided.
- If assets are released, the license, copyright information, and terms of use in the package should be provided. For popular datasets, paperswithcode.com/datasets has curated licenses for some datasets. Their licensing guide can help determine the license of a dataset.
- For existing datasets that are re-packaged, both the original license and the license of the derived asset (if it has changed) should be provided.
- If this information is not available online, the authors are encouraged to reach out to the asset's creators.

13. New assets

Question: Are new assets introduced in the paper well documented and is the documentation provided alongside the assets?

Answer: [\[Yes\]](#)

Justification: Yes, the data asset that we release is well documented and documentation is provided alongside the asset.

Guidelines:

- The answer NA means that the paper does not release new assets.
- Researchers should communicate the details of the dataset/code/model as part of their submissions via structured templates. This includes details about training, license, limitations, etc.
- The paper should discuss whether and how consent was obtained from people whose asset is used.
- At submission time, remember to anonymize your assets (if applicable). You can either create an anonymized URL or include an anonymized zip file.

14. Crowdsourcing and research with human subjects

Question: For crowdsourcing experiments and research with human subjects, does the paper include the full text of instructions given to participants and screenshots, if applicable, as well as details about compensation (if any)?

Answer: [\[NA\]](#)

Justification: We do not perform any crowd funding or research with human subjects in this research.

Guidelines:

- The answer NA means that the paper does not involve crowdsourcing nor research with human subjects.
- Including this information in the supplemental material is fine, but if the main contribution of the paper involves human subjects, then as much detail as possible should be included in the main paper.
- According to the NeurIPS Code of Ethics, workers involved in data collection, curation, or other labor should be paid at least the minimum wage in the country of the data collector.

15. Institutional review board (IRB) approvals or equivalent for research with human subjects

Question: Does the paper describe potential risks incurred by study participants, whether such risks were disclosed to the subjects, and whether Institutional Review Board (IRB) approvals (or an equivalent approval/review based on the requirements of your country or institution) were obtained?

Answer: [\[NA\]](#)

Justification: The answer NA means that the paper does not involve crowdsourcing nor research with human subjects.

Guidelines:

- The answer NA means that the paper does not involve crowdsourcing nor research with human subjects.
- Depending on the country in which research is conducted, IRB approval (or equivalent) may be required for any human subjects research. If you obtained IRB approval, you should clearly state this in the paper.
- We recognize that the procedures for this may vary significantly between institutions and locations, and we expect authors to adhere to the NeurIPS Code of Ethics and the guidelines for their institution.
- For initial submissions, do not include any information that would break anonymity (if applicable), such as the institution conducting the review.

16. Declaration of LLM usage

Question: Does the paper describe the usage of LLMs if it is an important, original, or non-standard component of the core methods in this research? Note that if the LLM is used only for writing, editing, or formatting purposes and does not impact the core methodology, scientific rigorousness, or originality of the research, declaration is not required.

Answer: [NA]

Justification: This paper does not involve using LLMs as important, original, or non-standard components.

Guidelines:

- The answer NA means that the core method development in this research does not involve LLMs as any important, original, or non-standard components.
- Please refer to our LLM policy (<https://neurips.cc/Conferences/2025/LLM>) for what should or should not be described.