scGeneScope: A Treatment-Matched Single Cell Imaging and Transcriptomics Dataset and Benchmark for Treatment Response Modeling

Joel Dapello[1]*, Marcel Nassar[1]*, Ridvan Eksi[1], Ban Wang[1], Jules Gagnon-Marchand[1], Kenneth T Gao[2†], Akram Baharlouei[1], Kyra Thrush-Evensen[1], Nina Riehs[1], Amy F Peterson[1], Aniket Tolpadi[1], Abhejit Rajagopal[3†], Henry E Miller[4†], Ashley Mae Conard[5], David Alvarez-Melis[5], Rory Stark[1], Simone Bianco[1], Morgan Levine[1], Ava P Amini[5], Alex Xijie Lu[5], Nicolo Fusi[5], Ravi Pandya[1], Valentina Pedoia[1], Hana El-Samad[1]

[1]Altos Labs [2]Genentech [3]Allen Institute [4]Shift Bioscience [5]Microsoft Research

Abstract

Understanding cellular responses to chemical interventions is critical to the discovery of effective therapeutics. Because individual biological techniques often measure only one axis of cellular response at a time, high-quality multimodal datasets are needed to unlock a holistic understanding of how cells respond to treatments and to advance computational methods that integrate modalities. However, many techniques destroy cells and thus preclude paired measurements, and attempts to match disparate unimodal datasets are often confounded by data being generated in incompatible experimental settings. Here we introduce scGeneScope, a multimodal single-cell RNA sequencing (scRNA-seq) and Cell Painting microscopy image dataset conditionally paired by chemical treatment, designed to facilitate the development and benchmarking of unimodal, multimodal, and multiple profile machine learning methods for cellular profiling. 28 chemicals, each acting on distinct biological pathways or *mechanisms of action (MoAs)*, were applied to U2-OS cells in two experimental data generation rounds, creating paired sets of replicates that were then profiled independently by scRNA-seq or Cell Painting. Using scGeneScope, we derive a replicate- and experiment-split treatment identification benchmark simulating MoA discovery under realistic laboratory variability conditions and evaluate unimodal, multimodal, and multiprofile models ranging in complexity from linear approaches to recent foundation models. Multiprofile integration improved performance in both the unimodal and multimodal settings, with gains more consistent in the former. Evaluation of unimodal models for MoA identification demonstrated that recent scRNA-seq foundation models deployed zero-shot were consistently outperformed by classic fit-to-data methods, underscoring the need for careful, realistic benchmarking in machine learning for biology. We release the scGeneScope dataset and benchmarking code to support further research.

1 Introduction

Measuring cellular responses to interventions is essential for understanding biological processes and developing effective treatments for disease. For example, biological screens to find potential

*Lead authors. Equal contribution. Correspondence to: {jdapello, mnassar}@altoslabs.com.
†Work done while at Altos Labs.

39th Conference on Neural Information Processing Systems (NeurIPS 2025) Track on Datasets and Benchmarks.

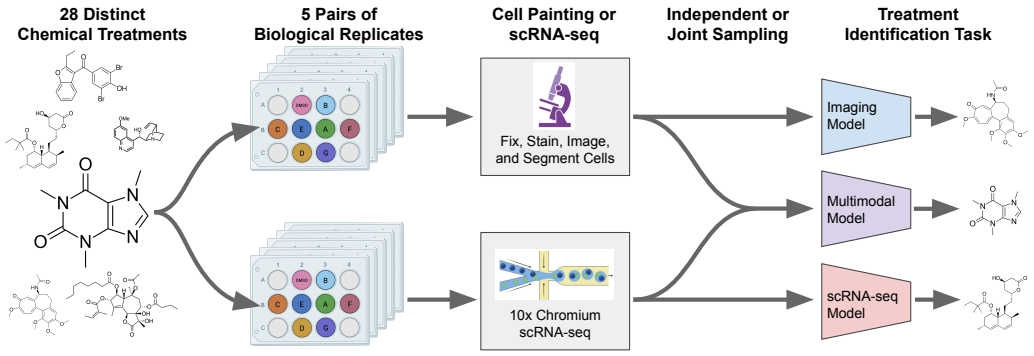

Figure 1: **The scGeneScope dataset and treatment identification benchmark.** 28 chemical treatments with diverse MoAs were applied to U2-OS cells in five pairs of biological replicates generated in two rounds. Replicate pairs were sent for independent Cell Painting and scRNA-seq profiling. During model training and benchmarking, single measurement profiles or sets of profiles from Cell Painting, scRNA-seq, or both were sampled uniformly by treatment category and fed into single or multiprofile imaging, scRNA-seq, or multimodal models trained for treatment classification.

new drugs often rely on the identification of changes in a cell's observable characteristics (i.e., its phenotype) after treatment with various chemical compounds. Understanding how a treatment works, known as its mechanism-of-action (MoA), is similarly critical to the drug development process [1, 2]. How a cell's response to an intervention is measured is a key design choice in obtaining accurate insights. Two common – but complementary – ways to profile cellular responses include transcriptomic technologies like single-cell RNA sequencing (scRNA-seq), which measure gene expression levels in individual cells [3], and microscopy assays like Cell Painting [1], which broadly illuminate cellular substructures and measure their physical structure and shape, known as cell morphology. Neither data modality alone is comprehensive: distinct transcriptomic states can yield similar morphologies, and meaningful transcriptomic shifts may lack detectable imaging changes [4].

Comparing or combining data from distinct measurement modalities such as transcriptomics and Cell Painting has been proposed as a means to provide richer information about how treatments affect cells [5, 4]. Unfortunately, this remains challenging to evaluate because RNA sequencing is a destructive measurement technique, and cannot be combined with Cell Painting on the same physical cells. A practical compromise is to match cell samples based on the treatment applied, wherein two biological samples, treated with the same chemical compound, are measured with separate measurement techniques. These replicate samples (or "replicates") should have similar patterns of underlying cellular response. To date however, treatment-matched datasets only include aggregate, bulk gene expression measurements [5, 4], which fail to capture the important variability between individual cells. Furthermore, existing datasets generally rely on post-hoc pairings from different experiments under different conditions, which can introduce unaccounted sources of variance and obfuscate analyses [5]. These limitations preclude direct comparisons between different Cell Painting and scRNA-seq measurement methods or evaluation of the practical benefits of integrating multiple types of biological data. Ultimately, this data gap hinders the development and comparison of unimodal, multimodal, and multiprofile methods for modeling cellular treatment response in a way that can meaningfully advance drug discovery.

Here we present scGeneScope, a high-quality, treatment-matched dataset of scRNA-seq and Cell Painting microscopy data. The scGeneScope dataset contains 627,704 scRNA-seq gene expression profiles and 716,767 single cell images of U2-OS cells perturbed with each of 28 chemical treatments spanning diverse MoAs. Unlike existing treatment-matched datasets, scGeneScope is collected through a series of tightly controlled parallel experiments with documented sources of variability, where scRNA-seq and Cell Painting assays are conducted on replicates treated with the same chemical treatments and over two independent rounds of experimental data generation. Using these data, we define the scGeneScope Treatment Identification Benchmark, with both Within Experiment evaluations and Held-out Experiment evaluations designed to simulate MoA identification and explore model generalization to independent experiments under realistic laboratory variability conditions.

We use the scGeneScope benchmark to evaluate a suite of classical fit-to-data methods as well as zero-shot foundation models in unimodal imaging and scRNA-seq settings, as well as a variety of late-fusion multimodal and multiprofile integration approaches to improve treatment identification based on imaging and scRNA-seq data (Figure 1). Our contributions include:

- **scGeneScope Dataset**: 627,704 scRNA-seq gene expression profiles and 716,767 Cell Painting single cell image crops of U2-OS cells perturbed with 28 chemical treatments, collected in five paired replicates across two independent experimental rounds with documented sources of variability (Datasheet B).
- **scGeneScope Treatment Identification Benchmark**: replicate- and experiment-aware splits defining *Within Experiment* and *Held-out Experiment* evaluations supporting unimodal, multimodal, and multiprofile model evaluation.
- **Extensive model evaluation**: comparison of classic baselines, zero-shot vision and scRNA-seq foundation models, plus late-fusion multimodal and multiprofile integration methods.

We illustrate the value of scGeneScope through a series of model evaluations, finding that, among models tested: (i) scRNA-seq models exhibit greater stability across experiments and generally outperform imaging models; (ii) within scRNA-seq models, classic approaches like scVI [6] outperform all tested foundation models used with zero-shot embeddings; (iii) while multiprofile methods consistently improve treatment identification performance in both unimodal and multimodal settings, multimodal integration methods had a relatively small impact and were inconsistent across evaluations. These results highlight that the promise of these models is yet to be fulfilled, and that fit-for purpose datasets and thoughtful approaches to model benchmarking are necessary to move machine learning for treatment identification forward from proof of concept to useful application.

2 Related Works

Datasets While there are many unpaired unimodal datasets for both scRNA-seq and Cell Painting, there is, to our knowledge, no public resource for treatment-matched data across these modalities. Large imaging-only repositories such as JUMP-Cell Painting [7], RxRx3 [8], and the benchmark of Moshkov *et al.* [9] accelerate morphology-based machine learning yet provide no transcriptomic layer. Meanwhile, resources like the CELLxGENE Census [10] or the Human Cell Atlas [11] contain large volumes of scRNA-seq data, but no high content imaging to relate higher order phenotypes. In terms of paired resources, Cell Painting has been paired with bulk gene expression assays in several toxicological screens [5, 12, 13, 4], but crucially no public resource couples Cell Painting with scRNA-seq for the same chemical treatments. Alternative multimodal efforts, including Image-seq [14] and bright-field imaging with scRNA-seq quantification for cell-size regulation [15], demonstrate ways to link imaging readouts to single-cell transcriptomics but are not paired with Cell Painting across interventions. There is thus a need for a high-quality, publicly-available dataset that simultaneously captures cellular morphology and single-cell gene expression response under matched chemical treatments. For a tabular breakdown of existing unimodal data resources, see Supplemental Section A.6.

Machine Learning Strategies for Phenotypic Data Analysis The analysis of both scRNA-seq and Cell Painting data is challenging, as biological signal may not be immediately accessible in raw data. To address this challenge, numerous computational methods have been proposed for both modalities. In transcriptomics, techniques like PCA and variational autoencoders (e.g., scVI [6]) have simplified and enhanced various downstream analyses by learning lower-dimensional representations of the data. More recently, natural language inspired, transformer-style foundation models for scRNA-seq have promised general purpose cell representations learned from millions of single-cell gene expression profiles [16–18]. However, simple baselines such as PCA or scVI fit to training data are observed to still outperform embeddings from these foundation models used in zero-shot settings [19, 20]. As such, the real-world utility of these recent models for key drug discovery tasks, like treatment prediction given a gene expression profile, remains unclear.

In microscopy, and Cell Painting in particular, computer vision techniques are used to extract lower-dimensional representations that measure the phenotype of cells. Classically, expert-defined features have been used, such as those implemented by the CellProfiler software package [21]. More recently, deep learning strategies for representation learning have been proposed. Initial approaches used

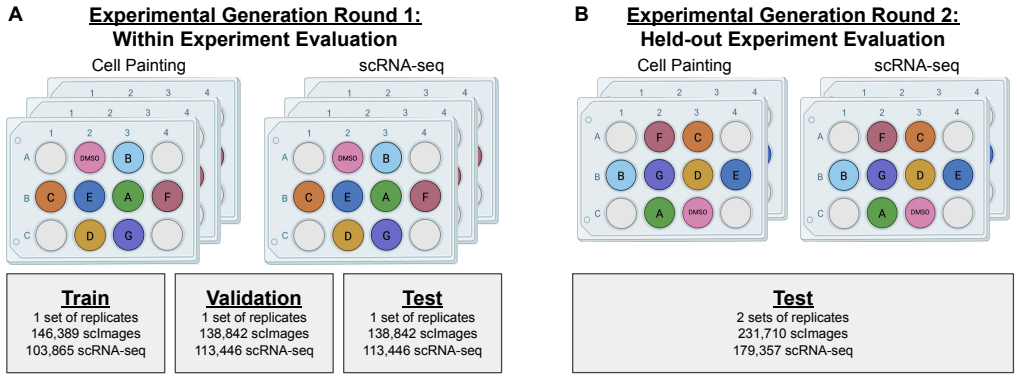

Figure 2: **scGeneScope dataset replicate and experiment splitting procedure for the treatment identification benchmark. (A)** Experimental data generation round 1 serves as a within experiment benchmark and is divided into train, validation, and test splits with one pair of replicates per split. **(B)** Experimental data generation round 2 serves as a held-out experiment benchmark.

off-the-shelf models pretrained on natural image datasets (e.g., ImageNet), which demonstrated performance on-par with, if not exceeding, expert-designed features [22, 23]. More recently, self-supervised models – implementing masked autoencoder (MAE) [24] and DINO-like [25] architectures and trained on microscopy datasets – have shown promising scaling properties and progressive improvement of morphological phenotype mapping with respect to model size and dataset quality.

3 scGeneScope Dataset and Benchmark

3.1 scGeneScope Data Generation

Our goal for data generation was to produce a dataset where each treatment produced a biologically distinct effect, to make classification well-posed. This can be challenging because distinct chemical compounds can often induce similar responses in cells and lead to similar phenotypes, creating a setting where each treatment can share effects with multiple others between modalities. To prevent this, we nominated 28 chemical compounds with biologically distinct MoAs. Section A.1.2 provides the full list of chemicals and further detail on the selection process. Over two rounds of experimental data generation, these compounds were applied to U2-OS cells, a widely studied bone cancer cell line [26, 27]. In addition to treating the cells with compounds, we also collected negative controls where cells were treated with dimethyl sulfoxide (DMSO), the solvent used for all compounds.

Replicates are defined as two samples from the same biological population, which are treated in the same way yet may be measured with separate measurement techniques. During the first round of data generation, we collected three sets of two replicates each, a complete set of experiments over all 28 compounds plus the negative control (Figure 2A). One replicate in a set is profiled with Cell Painting microscopy and the other with 10x Chromium scRNA-seq. In the second round of data generation, we likewise collected an additional two sets of replicates (Figure 2B). We note that each data generation round collected data under slightly different but functionally equivalent experimental procedures, capturing technical variation expected in real biological applications. See Supplemental Figures 3 and 4 as well as Section A.1.1 for a detailed description of replicates.

scRNA-seq data were processed into vectors of gene expression counts per single cell. Cell Painting images were processed into single cell crops, which we refer to as scImages. Section A.1 provides the full details, numbers, and protocols on data generation and post processing. From these experiments, we collected $627,704$ scRNA-seq profiles and $716,767$ scImages.

3.2 scGeneScope Treatment Identification Benchmark

With the scGeneScope datasets, we define a downstream task of *treatment identification*. Here, models can be evaluated based on their performance in classifying which of the 28 chemical compound treatments cell(s) were treated with, given access to either scRNA-seq data, scImage data, or both.

Because scGeneScope was designed such that each treatment represented a distinct MoA, our treatment identification task effectively simulates how strongly models will perform at MoA prediction, which is critical to drug screening applications as it identifies how chemical compounds act upon cells [28, 29]. We evaluate classification performance using balanced accuracy and macro-F1 metrics, as these metrics are robust to class imbalance – although we include equal numbers of replicates for each of the 28 chemical compounds, each replicate may generate different numbers of single cells, and thus there may be different data densities across the different treatments. For a visualization of single cell counts per treatment and replicate, see Supplemental Figure 5.

3.3 Input Settings for Modeling

To facilitate methods development, we organize scGeneScope into four modeling input settings based upon combinations of two variables: whether models are given *unimodal* or *multimodal* profiles, and whether models are given *single profiles* or *multiple profiles (multiprofile)*.

In the unimodal versus multimodal setting, models are respectively given one modality alone versus both modalities. Comparing models across these two settings enables users of scGeneScope to evaluate if multimodal models outperform unimodal models. Additionally, unimodal models can be given access to either scRNA-seq or scImage data. While we do not expect fully equal performance as some MoAs may have stronger signal in one modality versus the other, this still permits the comparison of models for each modality on a standardized problem, allowing users of scGeneScope to evaluate the effectiveness of current transcriptomic versus imaging methods for drug screening applications. Specifically, we have:

- **Unimodal profiles**: each input example is a tuple of the form (x_i, t_i) where x_i represents single cell profile(s) from either scRNA-seq or scImages and t_i represents the one-hot encoding of the chemical compound treatment for the sample.
- **Multimodal profiles**: each input example is a tuple of the form $(x_i^{scRNA}, x_j^{scImage}, t_i)$ where x_i^{scRNA} and $x_j^{scImage}$ represent scRNA-seq and scImage data, respectively, for a given treatment t_i. The two data points share a treatment ($t_i = t_j$) and have been randomly paired.

In the single profile versus multiprofile setting, models are respectively given input examples including either one profile per modality versus k profiles per modality. Even within the same experiment, single cells treated with the same chemical compound may exhibit a range of responses to the compound [30]. In principle, this can make single profile classification more challenging, as models may be presented with cells with low or no response. However, an open question for multiple instance machine learning methods is how to best integrate information from multiple cell profiles [31, 32]. Hence, this setting enables users of scGeneScope to evaluate if multiple instance methods are performant by quantifying improvements over single instance models, and against other multiple instance methods. Specifically, we have:

- **Single profile**: each input example x_i is a vector $[1 \times D]$ where D is the dimension of the single cell profile.
- **Multiprofile**: each input example x_i is a matrix $[k \times D]$ where D is the dimension of the single cell profile, and k is the number of profiles, sampled uniformly without replacement within a given treatment. In reported results, $k = 32$ (See Appendix A.7 for different k values).

Combining these two setting variables, we produce four sets of benchmarking conditions: unimodal single profile, unimodal multiprofile, multimodal single profile, and multimodal multiprofile.

3.4 Data Splits

We next produce train, validation, and test splits for the scGeneScope dataset. Because drug screening experiments can scale to millions of individual measurements [7], it is infeasible to conduct all the data in a single round of measurement. However, measurements from different rounds may be collected under slightly different conditions that create batch effects in the resulting single-cell profiles, which can cause downstream machine learning and data analysis pipelines to overfit [33].

We therefore provide two test datasets of differential difficulty: the easier, *Within Experiment (WE)* setting that is closer in-distribution relative to the train and validation splits (Figure 2A) and the more difficult, *Held-out Experiment (HE)* setting that reflects realistic technical variation from an independent experiment (Figure 2B). The HE setting more closely simulates real drug screening applications, as it reproduces the technical differences commonly present in these applications (Supplemental Figure 3).

One set of replicates from the first data generation round is used for each of the train, validation, and WE test datasets (Figure 2A). All replicates from the second data generation round are used to construct the HE test dataset (Figure 2B). Under this strategy, the train set has $146,389$ scImages and $103,865$ scRNA-seq profiles, the validation set has $138,842$ scImages and $113,446$ scRNA-seq profiles, the WE test set has $138,842$ scImages and $113,446$ scRNA-seq profiles, and the HE test set has $231,710$ scImages and $179,357$ scRNA-seq profiles. For each of our four model input settings, we sample scRNA-seq and scImage single cell profiles at uniform probability across the 28 treatment classes.

4 Benchmarked Models

We decompose our treatment identification benchmark task into three steps: feature extraction, pooling and integration, and classification. In feature extraction, raw single cell profiles are transformed into a lower dimensional representation, which ideally contains latent biological signal. In pooling and integration, if the input contains multiple profiles or multiple modalities, these are combined into a single vector representation. In classification, the final representation is used to predict the treatment class. Each of these steps has been previously explored in both scRNA-seq and imaging data analysis, and hence, we benchmark a suite of established methods, which we detail below.

4.1 Feature Extraction Models

Imaging Models: Following prior works that have demonstrated that ImageNet-trained models, even without fine-tuning, can extract strong representations of Cell Painting images [22], we evaluate a variety of ImageNet-trained baseline models [34], loaded via the `torchvision` library [35], including two convolutional networks: ResNet-50 and ResNet-152 [36], and two transformer architectures: ViT-L/16 and ViT-H/14 [37]. Additionally, we evaluate ResNet-50 and ViT-H trained with Contrastive Language Image Pre-training (CLIP) [38]. Finally, we include OpenPhenom-S/16 [39], a channel-agnostic foundation model for microscopy which was pretrained using a Masked Autoencoder (MAE) task on over 3 million images from the RxRx3 [8] and JUMP-CP [40] collections. Following previous practice for feature extraction from Cell Painting images [23, 22, 41], all five Cell Painting stain channels were passed through each imaging model independently, embeddings for each channel were produced, and then the embeddings were concatenated together. For model specific details, see Supplemental Section A.2.1.

scRNA-seq models: We evaluate a number of scRNA-seq feature extraction models, falling into two categories. First, we consider classic statistical algorithms and machine learning methods directly fit to a dataset of interest, which we fit to the train split. We fit PCA (2000 components, selected via cross validation) and scVI [6] (200 latent dimensions) to the train data. Second, we consider recently-proposed single-cell foundation models that have been pre-trained on large scRNA-seq datasets. We analyze scGPT [16], Geneformer [17], Universal Cell Embeddings (UCE, 4-layer model) [18], scVI-1 [42], and scVI-2 [43]. Following the claim of these works that single cell foundation models learn transferrable representations of single cell biology, no finetuning of the embedding weights is performed – the data is embedded according to each specific model's recommended procedure. For model specific details, see Supplemental Section A.2.2.

4.2 Aggregation and Integration Models

In our multiple profile input setting, models receive multiple single cell profiles for each modality they are given. In our multimodal input setting, models receive profiles from both scRNA-seq and scImages. Before classification, these multiple profiles must be combined into a single vector representation. We refer to combining multiple profiles as "aggregation" and combining multiple modalities as "integration".

Aggregation: We evaluate average pooling (i.e., averaging all input single cell profiles), as well as permutation-invariant neural networks for combining sets, DeepSet [44] and Set Transformers [45]. DeepSets and Set Transformers are two examples of permutation invariant and equivariant neural networks designed to operate on sets, where the ordering of elements is inherently arbitrary. DeepSet introduced a foundational framework for processing such data, showing that any permutation invariant function over a set can be decomposed into a function of summed element-wise transformations: $\rho(\sum_i \phi(x_i))$. This architecture ensures invariance by construction and has been widely adopted in domains ranging from point cloud processing to graph learning. Set Transformers build upon this by incorporating attention mechanisms to capture pairwise and higher-order interactions between elements while maintaining permutation equivariance at the layer level and overall permutation invariance at the output. A subtle but important difference between Set Transformers and vanilla Transformers is the absence of positional encodings, which are unnecessary (and inappropriate) for unordered inputs. Unlike DeepSet, which aggregate early, Set Transformers allow richer inter-element reasoning via self-attention, making them more expressive for tasks where relational structure within sets is important.

Integration: We evaluate the late fusion strategy of concatenating representations from scRNA-seq and scImages. While there is extensive literature on multimodal integration [46], we focus on late fusion because many of our feature extraction methods require either frozen or non-differentiable models, though in principle the scGeneScope benchmark enables evaluation of early, mid, and late fusion methods.

4.3 Classification

After representations have been extracted by feature extraction methods, and if necessary, pooled and/or integrated into a single vector representation, we fit task heads to classify treatment. For a given example with treatment label t_i, the task head $h : \mathbb{R}^{D'} \to \mathbb{R}^C$ maps the fused embedding to a probability distribution over C treatment classes. The model is trained to minimize the cross entropy loss: $\mathcal{L} = -\sum_{i=1}^{N} \sum_{c=1}^{C} y_{i,c} \log(\hat{y}_{i,c})$ where $y_{i,c}$ is the ground truth indicator for treatment c and $\hat{y}_{i,c}$ is the predicted probability for treatment c for example i. The task head hyperparameters are selected through hyperparameter optimization on the train and validation splits (Section A.2.3). Final models are trained on the training splits with early stopping monitored on the validation splits. All reported scores are the mean and standard error over 5 training seeds.

5 Results

For all models and settings we evaluated, the WE and HE treatment identification scores can be found in Table 1 and scores are visualized in Supplemental Figure 6.

5.1 Benchmarking Unimodal Single Profile Imaging and scRNA-seq Models

We started by benchmarking models in the unimodal single profile setting, where each imaging or scRNA-seq model makes predictions on individual scImage or scRNA-seq profiles, respectively.

Among imaging models we observe relatively similar scores across all supervised ImageNet pre-trained models we tested, with no clear trends emerging based on model size or architecture. Transformer-based models appear to slightly outperform convolutional models, with ViT-L being the best performer on the WE evaluation and ViT-H generalizing best to the HE evaluation. No imaging models are robust to the HE evaluation setting, with all models losing $5\% - 7\%$ balanced accuracy when evaluated on HE splits. Interestingly, CLIP pre-trained models consistently perform worse than their supervised counterparts. Finally, OpenPhenom, the only domain specific imaging foundation model, performs notably worse than any other imaging model we tested, highlighting the challenge of creating robust imaging foundation models that generalize to different cellular microscopy datasets.

Among scRNA-seq models we observe the highest performance across both the WE and HE splits from scVI fit directly to the training splits. PCA also scores highly on the WE evaluation, but drops substantially on the HE evaluation. Among the pretrained scRNA-seq foundation models we tested, scGPT performs the highest on the WE evaluation and also has the most stable generalization to

Table 1: **scGeneScope Treatment Identification Benchmark results for all models evaluated.** Scores are reported for Within Experiment (WE) and Held-out Experiment (HE) test splits (mean ± standard error over five training seeds). Models marked with † indicate zero-shot embeddings with the encoders not fit to data, while models marked with ‡ use encoders fit directly to the train data only. For a graphical display of benchmarking results, see Supplemental Figure 6.

	Within-Experiment Test		**Held-out-Experiment Test**	
Model	**Bal. Acc.**	**Macro-F1**	**Bal. Acc.**	**Macro-F1**
Imaging models (Unimodal Single Profile)				
ResNet-50†	0.2623 ± 0.0011	0.2434 ± 0.0010	0.2127 ± 0.0024	0.1870 ± 0.0036
ResNet-152†	0.2567 ± 0.0133	0.2363 ± 0.0126	0.2109 ± 0.0122	0.1898 ± 0.0131
ViT-L†	**0.2790 ± 0.0085**	**0.2642 ± 0.0125**	0.2078 ± 0.0025	0.1836 ± 0.0020
ViT-H†	0.2673 ± 0.0017	0.2482 ± 0.0048	**0.2235 ± 0.0145**	**0.1947 ± 0.0126**
ResNet-50-CLIP†	0.2239 ± 0.0066	0.1925 ± 0.0172	0.1982 ± 0.0182	0.1618 ± 0.0168
ViT-H-CLIP†	0.2033 ± 0.0237	0.1506 ± 0.0273	0.1644 ± 0.0274	0.1071 ± 0.0331
OpenPhenom†	0.1681 ± 0.0125	0.1402 ± 0.0109	0.1420 ± 0.0085	0.1196 ± 0.0121
scRNA-seq models (Unimodal Single Profile)				
PCA‡	0.5103 ± 0.0059	0.4734 ± 0.0104	0.2072 ± 0.0477	0.2031 ± 0.0557
scVI‡	**0.5172 ± 0.0039**	**0.4968 ± 0.0042**	**0.5161 ± 0.0054**	**0.4661 ± 0.0054**
scVI-1†	0.3190 ± 0.0063	0.2895 ± 0.0031	0.2799 ± 0.0050	0.2197 ± 0.0040
scVI-2†	0.2184 ± 0.0020	0.1915 ± 0.0017	0.2108 ± 0.0046	0.1558 ± 0.0022
UCE†	0.2232 ± 0.0027	0.2036 ± 0.0061	0.2018 ± 0.0044	0.1455 ± 0.0059
Geneformer†	0.2805 ± 0.0051	0.2481 ± 0.0045	0.1959 ± 0.0087	0.1218 ± 0.0068
scGPT†	0.3870 ± 0.0036	0.3780 ± 0.0033	0.3807 ± 0.0036	0.3408 ± 0.0053
Imaging models (Unimodal Multiprofile)				
AvgPool + ViT-L†	**0.4711 ± 0.0089**	**0.4545 ± 0.0072**	0.2575 ± 0.0088	0.2255 ± 0.0107
AvgPool + ViT-H†	0.3870 ± 0.0108	0.3743 ± 0.0075	**0.2729 ± 0.0067**	**0.2192 ± 0.0063**
DeepSet + ViT-L†	0.4362 ± 0.0070	0.4306 ± 0.0091	0.2502 ± 0.0081	0.2094 ± 0.0059
DeepSet + ViT-H†	0.3525 ± 0.0170	0.3497 ± 0.0216	0.2028 ± 0.0223	0.1739 ± 0.0214
Transformer + ViT-L†	0.3297 ± 0.0277	0.2848 ± 0.0365	0.1590 ± 0.0072	0.1502 ± 0.0168
Transformer + ViT-H†	0.3160 ± 0.0249	0.2932 ± 0.0264	0.2193 ± 0.0415	0.1812 ± 0.0223
scRNA-seq models (Unimodal Multiprofile)				
AvgPool + PCA‡	0.5993 ± 0.0053	0.5564 ± 0.0100	0.1788 ± 0.0242	0.1636 ± 0.0258
AvgPool + scVI‡	0.6321 ± 0.0307	0.6156 ± 0.0295	0.5739 ± 0.0246	0.5480 ± 0.0286
AvgPool + scGPT†	0.4548 ± 0.0105	0.4614 ± 0.0099	0.3161 ± 0.0087	0.2761 ± 0.0098
DeepSet + PCA‡	0.5847 ± 0.0138	0.5284 ± 0.0112	0.4299 ± 0.0367	0.3936 ± 0.0398
DeepSet + scVI‡	**0.6373 ± 0.0225**	**0.6323 ± 0.0164**	**0.5809 ± 0.0398**	**0.5343 ± 0.0450**
DeepSet + scGPT†	0.4204 ± 0.0141	0.4282 ± 0.0119	0.3387 ± 0.0101	0.3104 ± 0.0089
Transformer + PCA‡	0.4475 ± 0.0249	0.4241 ± 0.0269	0.2580 ± 0.0882	0.2301 ± 0.0813
Transformer + scVI‡	0.6154 ± 0.0173	0.6044 ± 0.0169	0.4561 ± 0.0443	0.4473 ± 0.0437
Transformer + scGPT†	0.4502 ± 0.0061	0.4540 ± 0.0060	0.3328 ± 0.0083	0.2972 ± 0.0056
Multimodal Single Profile per Modality				
PCA‡ + ViT-L†	0.5099 ± 0.0041	0.4960 ± 0.0045	0.3573 ± 0.0060	0.3390 ± 0.0055
scVI‡ + ViT-L†	0.4995 ± 0.0078	0.4862 ± 0.0052	**0.5172 ± 0.0071**	**0.5088 ± 0.0075**
scGPT† + ViT-L†	0.4132 ± 0.0012	0.4114 ± 0.0012	0.4181 ± 0.0027	0.4153 ± 0.0018
PCA‡ + ViT-H†	**0.5167 ± 0.0026**	**0.5011 ± 0.0022**	0.3577 ± 0.0104	0.3370 ± 0.0065
scVI‡ + ViT-H†	0.4923 ± 0.0123	0.4829 ± 0.0030	0.5247 ± 0.0082	0.5111 ± 0.0149
scGPT† + ViT-H†	0.3994 ± 0.0009	0.3992 ± 0.0022	0.4307 ± 0.0025	0.4223 ± 0.0011
Multimodal Multiple Profiles per Modality				
AvgPool + PCA‡ + ViT-L†	0.4659 ± 0.0157	0.4397 ± 0.0146	0.3278 ± 0.0310	0.2884 ± 0.0368
AvgPool + scVI‡ + ViT-L†	**0.6815 ± 0.0102**	**0.6590 ± 0.0088**	0.5442 ± 0.0189	0.5401 ± 0.0156
AvgPool + scGPT† + ViT-L†	0.4728 ± 0.0181	0.4569 ± 0.0108	0.3741 ± 0.0260	0.3250 ± 0.0225
AvgPool + PCA‡ + ViT-H†	0.4947 ± 0.0145	0.4749 ± 0.0197	0.2942 ± 0.0721	0.2495 ± 0.0741
AvgPool + scVI‡ + ViT-H†	0.6444 ± 0.0305	0.6342 ± 0.0262	**0.5677 ± 0.0249**	**0.5512 ± 0.0287**
AvgPool + scGPT† + ViT-H†	0.4204 ± 0.0090	0.3773 ± 0.0167	0.4258 ± 0.0233	0.3840 ± 0.0202

the HE evaluation, but remains more than 10% below scVI fit exclusively to the training data. The other scRNA-seq foundation models (UCE and Geneformer) we tested underperformed PCA in our unimodal single profile evaluation, and hence we focus on scGPT for the other scRNA-seq foundation model evaluation scenarios.

Comparing across imaging and scRNA-seq models we tested, the two scRNA-seq models capable of maintaining performance on the HE evaluation (scVI fit-to-data and pretrained scGPT) outperform the best imaging models we tested, both in terms of absolute scores and also in terms of stability when generalizing from WE to HE evaluations. We note that while we did not directly balance the number of scImages versus scRNA-seq profiles in the train sets, there are approximately 40% more scImage profiles than scRNA-seq profiles to learn from (Section 3.4), suggesting that lack of training examples do not account for the disparity in performance.

5.2 Benchmarking Conditional Multimodal Integration Models

To investigate the potential of integrating measurements across imaging and scRNA-seq modalities to derive a more holistic representation of cell response to drug treatment, we next integrated the overall best-performing methods from our unimodal evaluations. We integrated two imaging models (ViT-L and ViT-H) with three scRNA-seq models (PCA and scVI fit to the train set, and pretrained scGPT) and explored the simple late-stage conditional multimodal fusion strategy described in Section 4.2 for all crosses of the models listed.

Interestingly, we do not observe consistent improvements in performance using late-stage multimodal integration across the different model pairings. For instance, scGPT paired with either imaging model shows modest but consistent improvements ($+3\%$ over scGPT alone for both the WE and HE evaluations), but scVI fit-to-data combined with ViT-L decreases or stays the same for the WE and HE evaluations, respectively, compared to scVI alone. Meanwhile, PCA paired with ViT-L scores the same for the WE evaluation, but generalizes substantially better to the HE evaluation. For detailed more detailed breakdowns of the unimodal and multimodal model classwise performance and error patterns, refer to Supplemental Section A.5.

5.3 Benchmarking Multiprofile Aggregation Methods

Finally, we sought to assess how unimodal and multimodal models perform as the number of single cell profiles per modality increased. To adapt the single profile unimodal and multimodal models benchmarked to a multiprofile setting, we explored three different multiprofile aggregation strategies described in Section 4.2: average pooling (AvgPool), DeepSet, and SetTransformers.

Across both the WE and HE evaluations, multiprofile aggregation improved performance of both unimodal imaging and scRNA-seq models. In nearly all cases (with the exception of scGPT on the HE evaluation), the best performing aggregation method, found through HPO on the validation set, outperformed the single profile version of the base model. We note that no tested method was completely stable across WE and HE evaluations. Among the aggregation methods tested, the AvgPool and DeepSet aggregators were consistently strong across both modalities, particularly on the WE evaluation. Interestingly, DeepSet showed more consistent generalization to the HE evaluation.

Finally, we took the AvgPool pooling method since it was generally reliable and high performing and applied it to the multimodal case in order to explore whether multiple-profile, multimodal aggregation outperformed the single-profile multimodal or multiprofile unimodal scenarios. Interestingly, while the AvgPool + scVI + ViT-L model performed the best of all models on the WE evaluation, its performance dropped on the HE evaluation, with no statistically significant difference from the multiprofile unimodal DeepSet or AvgPool + scVI models.

6 Discussion

In this work we introduce scGeneScope, a treatment-matched resource that pairs single cell RNA-sequencing data with Cell Painting images and provides treatment-identification benchmarks split by both replicate and experiment. Several observations emerged during the course of our benchmarking efforts. First, evaluating performance across both the Within Experiment (WE) and Held-out Experiment (HE) splits exposed where a number of models may be overfitting and failing

to generalize. All imaging models, many scRNA-seq models, and many fusion techniques built on top of them lost substantial accuracy when moving from the WE to the HE evaluations, highlighting the diagnostic value of experiment-level splits. Second, our results suggest that widely publicized foundation models are not yet plug-and-play solutions for cellular phenotyping. A modestly sized, data-specific scVI encoder trained from scratch matched or surpassed all zero-shot scRNA-seq transformers. Although there is no strong expectation that ImageNet-pretrained models perform well out-of-the-box, previous works have established that these models are capable of capturing cellular phenotypes zero-shot [22, 41], so it is surprising that these imaging models substantially underperformed the scRNA-seq models in our study on a related treatment identification task. Even more surprising, OpenPhenom, the only domain specific imaging model in our cohort, performed the worst of all models tested across both modalities. Third, integrating information across profiles and modalities often helped performance, but success was dependent on the details. Simple average pooling and DeepSet aggregators consistently improved unimodal results but were unstable across the WE and HE evaluations. Likewise, multimodal, multiple profile scVI + ViT-L produced the strongest scores on the WE evaluation, but did not generalize as well to the HE evaluation.

The observation that current models struggle on the scGeneScope benchmark is alarming when considering that our dataset with only 28 compounds in a single cell line pales in comparison to the real-word challenges of scaling to hundreds of thousands of perturbations, multiple doses, and many additional cell lines [47, 48]. We acknowledge that we have left batch integration strategies and classical CellProfiler-based imaging features underexplored. It would also be worthwhile to investigate finetuned versions of the large encoders we evaluated only in zero-shot mode, though we argue that having to finetune a foundation model or perform batch integration for each specific dataset and task reduces the utility as a universal embedding model, and encourage the field to push towards truly robust encoding methods. Likewise, the fusion strategies we tried were restricted to late aggregation; early fusion, cross-modal attention, and joint generative modeling remain untested and could provide more substantial gains.

Taken together, our results offer a reality check and present significant implications for current efforts to develop foundational models of cells [49]. Even on a seemingly tractable benchmark, biological foundation models and off-the-shelf fusion pipelines fall short of current expectations. We hope that making the scGeneScope data, code, and evaluation harness public will catalyze work on more appropriate pre-training objectives, the development of multiprofile and multimodal aggregation operators, and the adoption of replicate- and experiment-aware splits. Such efforts will help machine learning accelerate mechanism-of-action discovery and, ultimately, therapeutics development.

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.rxrx.ai/rxrx3`, 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] Shibla Abdulla, Brian D. Aevermann, Pedro Assis, Seve Badajoz, Sidney M. Bell, Emanuele Bezzi, Batuhan Cakir, Jim Chaffer, and Ambrose Carr. Cz cell$\times$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 Lundeberg, 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.

Stubbington, Fabian J. Theis, Mathias Uhlén, Alexander van Oudenaarden, Allon Wagner, Fiona Watt, Jonathan Weissman, Barbara Wold, Ramnik Xavier, Nir Yosef, and Human Cell Atlas Meeting Participants. The human cell atlas. *eLife*, 6:e27041, December 2017.

[12] Joshua A. Harrill, Logan J. Everett, Derik E. Haggard, Thomas Sheffield, Joseph L. Bundy, Clinton M. Willis, Russell S. Thomas, Imran Shah, and Richard S. Judson. High-throughput transcriptomics platform for screening environmental chemicals. *Toxicological Sciences*, 181(1):68–89, 2021.

[13] Jo Nyffeler, Clinton Willis, Felix R. Harris, M. J. Foster, Bryant Chambers, Megan Culbreth, Richard E. Brockway, Sarah Davidson– Fritz, Daniel Dawson, Imran Shah, Katie Paul Friedman, Dan Chang, Logan J. Everett, John F. Wambaugh, Grace Patlewicz, and Joshua A. Harrill. Application of cell painting for chemical hazard evaluation in support of screening-level chemical assessments. *Toxicology and Applied Pharmacology*, 468:116513, 2023.

[14] Christa Haase, Karin Gustafsson, Shenglin Mei, Shu-Chi Yeh, Dmitry Richter, Jelena Milosevic, Raphaël Turcotte, Peter V. Kharchenko, David B. Sykes, David T. Scadden, and Charles P. Lin. Image-seq: Spatially resolved single-cell sequencing guided by in situ imaging. *Nature Methods*, 19:1622–1633, 2022.

[15] Olivia Padovan-Merhar, Gautham P. Nair, Andrew G. Biaesch, Andreas Mayer, Steven Scarfone, Shawn W. Foley, Angela R. Wu, L. Stirling Churchman, Abhyudai Singh, and Arjun Raj. Single mammalian cells compensate for differences in cellular volume and dna copy number through independent global transcriptional mechanisms. *Molecular Cell*, 58(2):339–352, 2015.

[16] Haotian Cui, Chloe Wang, Hassaan Maan, Kuan Pang, Fengning Luo, Nan Duan, and Bo Wang. scgpt: Toward building a foundation model for single-cell multi-omics using generative ai. *Nature Methods*, 21:1470–1480, 2024.

[17] Christina V. Theodoris, Ling Xiao, Anant Chopra, Mark D. Chaffin, Zeina R. Al Sayed, Matthew C. Hill, Helene Mantineo, Elizabeth M. Brydon, Zexian Zeng, X. Shirley Liu, and Patrick T. Ellinor. Transfer learning enables predictions in network biology. *Nature*, 618(7965):616–624, 2023.

[18] Yanay Rosen, Yusuf Roohani, Ayush Agarwal, Leon Samotorcan, Stephen R. Quake, and Jure Leskovec. Universal cell embeddings: A foundation model for cell biology. *bioRxiv*, 2023.

[19] Rebecca Boiarsky, Nalini M. Singh, Alejandro Buendia, Ava P. Amini, Gad Getz, and David Sontag. Deeper evaluation of a single-cell foundation model. *Nature Machine Intelligence*, 6(12):1443–1446, 2024. Matters Arising; shows that logistic regression can outperform the zero-shot scBERT RNA-seq foundation model.

[20] Kasia Z. Kedzierska, Lorin Crawford, Ava P. Amini, and Alex X. Lu. Zero-shot evaluation reveals limitations of single-cell foundation models. *Genome Biology*, 26(1):101, 2025.

[21] Anne E. Carpenter, Thouis R. Jones, Michael R. Lamprecht, Colin Clarke, In Han Kang, Ola Friman, David A. Guertin, Joo Han Chang, Robert A. Lindquist, Jason Moffat, Polina Golland, and David M. Sabatini. Cellprofiler: image analysis software for identifying and quantifying cell phenotypes. *Genome Biology*, 7:R100, 2006.

[22] Nick Pawlowski, C. Caicedo Juan Shantanu Singh, E. Carpenter Anne and Amos Storkey. Automating morphological profiling with generic deep convolutional networks. *bioRxiv*, 2016. Posted 2 November 2016.

[23] D. Michael Ando, Y. McLean Cory and Marc Berndl. Improving phenotypic measurements in high-content imaging screens. *bioRxiv*, 2017. Posted 10 July 2017.

[24] Oren Kraus, Kian Kenyon-Dean, Saber Saberian, Maryam Fallah, Peter McLean, Jess Leung, Vasudev Sharma, Ayla Khan, Jia Balakrishnan, Safiye Celik, Dominique Beaini, Maciej Sypetkowski, Chi Vicky Cheng, Kristen Morse, Maureen Makes, Ben Mabey, and Berton Earnshaw. Masked autoencoders for microscopy are scalable learners of cellular biology. In *2024 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 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] Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, Gretchen Krueger, and Ilya Sutskever. Learning transferable visual models from natural language supervision. *arXiv preprint arXiv:2103.00020*, 2021.

[39] Recursion. Openphenom-s/16: A foundation model for microscopy data. `https://www.rxrx.ai/phenom`, November 2024.

[40] Mark A. Bray, Saga M. Gustafsdottir, Mohammad H. Rohban, Shantanu Singh, Vebjorn Ljosa, Kelly L. Sokolnicki, Joshua A. Bittker, Nicole E. Bodycombe, Vlado Dancík, Timothy P. Hasaka, Chantal S. Hon, Maria M. Kemp, Kan Li, Dilanthi Walpita, Mathias J. Wawer, Todd R. Golub, Stuart L. Schreiber, Paul A. Clemons, Alykhan F. Shamji, and Anne E. Carpenter. A dataset of images and morphological profiles of 30 000 small-molecule treatments using the cell painting assay. *GigaScience*, 6(12):1–5, Dec 2017.

[41] 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.

[42] Chan Zuckerberg CELL×GENE Team. scVI-1 pretrained model on CELL×GENE Census (release 2023-05-15). `https://cellxgene.cziscience.com/census-models`, 2023.

[43] Chan Zuckerberg CELL×GENE Team. scVI-2 pretrained model on CELL×GENE Census (release 2024-02-12). `https://cellxgene.cziscience.com/census-models`, 2024.

[44] Manzil Zaheer, Satwik Kottur, Siamak Ravanbakhsh, Barnabas Poczos, Russ R Salakhutdinov, and Alexander J Smola. Deep sets. In I. Guyon, U. Von Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett, editors, *Advances in Neural Information Processing Systems*, volume 30. Curran Associates, Inc., 2017.

[45] Juho Lee, Yoonho Lee, Jungtaek Kim, Adam Kosiorek, Seungjin Choi, and Yee Whye Teh. Set transformer: A framework for attention-based permutation-invariant neural networks. In Kamalika Chaudhuri and Ruslan Salakhutdinov, editors, *Proceedings of the 36th International Conference on Machine Learning*, volume 97 of *Proceedings of Machine Learning Research*, pages 3744–3753. PMLR, 09–15 Jun 2019.

[46] Ricard Argelaguet, Anna SE Cuomo, Oliver Stegle, and John C Marioni. Computational principles and challenges in single-cell data integration. *Nature biotechnology*, 39(10):1202–1215, 2021.

[47] Jesse Zhang, Airol A. Ubas, Richard de Borja, Valentine Svensson, Nicole Thomas, Neha Thakar, Ian Lai, Aidan Winters, Umair Khan, Matthew G. Jones, Vuong Tran, Joseph Pangallo, Efthymia Papalexi, Ajay Sapre, Hoai Nguyen, Oliver Sanderson, Maria Nigos, Olivia Kaplan, Sarah Schroeder, Bryan Hariadi, and Simone Marrujo. Tahoe-100m: A giga-scale single-cell perturbation atlas for context-dependent gene function and cellular modeling. *bioRxiv*, 2025.

[48] David S Fischer, Martin A Villanueva, Peter S Winter, and Alex K Shalek. Adapting systems biology to address the complexity of human disease in the single-cell era. *Nature Reviews Genetics*, pages 1–18, 2025.

[49] Charlotte Bunne, Yusuf Roohani, Yanay Rosen, Ankit Gupta, Xikun Zhang, Marcel Roed, Theo Alexandrov, Mohammed AlQuraishi, Patricia Brennan, Daniel B Burkhardt, et al. How to build the virtual cell with artificial intelligence: Priorities and opportunities. *Cell*, 187(25):7045–7063, 2024.

[50] Zaber Technologies Inc. Microscope autofocus with python and opencv. `https://github.com/zabertech/zaber-examples/tree/main/examples/microscope_autofocus`, 2024.

[51] Carsen Stringer, Michalis Michaelos, and Marius Pachitariu. Cellpose: a generalist algorithm for cellular segmentation. *Nature Methods*, 18(1):100–106, 2021.

[52] Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, Alban Desmaison, Andreas Köpf, Edward Z. Yang, Zach DeVito, Martin Raison, Alykhan Tejani, Sasank Chilamkurthy, Benoit Steiner, Lu Fang, Junjie Bai, and Soumith Chintala. Pytorch: An imperative style, high-performance deep learning library. *Advances in Neural Information Processing Systems*, abs/1912.01703, 2019.

[53] William Falcon and The PyTorch Lightning team. PyTorch Lightning, March 2019.

[54] Omry Yadan. Hydra - a framework for elegantly configuring complex applications. Github, 2019.

[55] F Alexander Wolf, Philipp Angerer, and Fabian J Theis. Scanpy: large-scale single-cell gene expression data analysis. *Genome biology*, 19:1–5, 2018.

A Technical Appendices and Supplemental 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

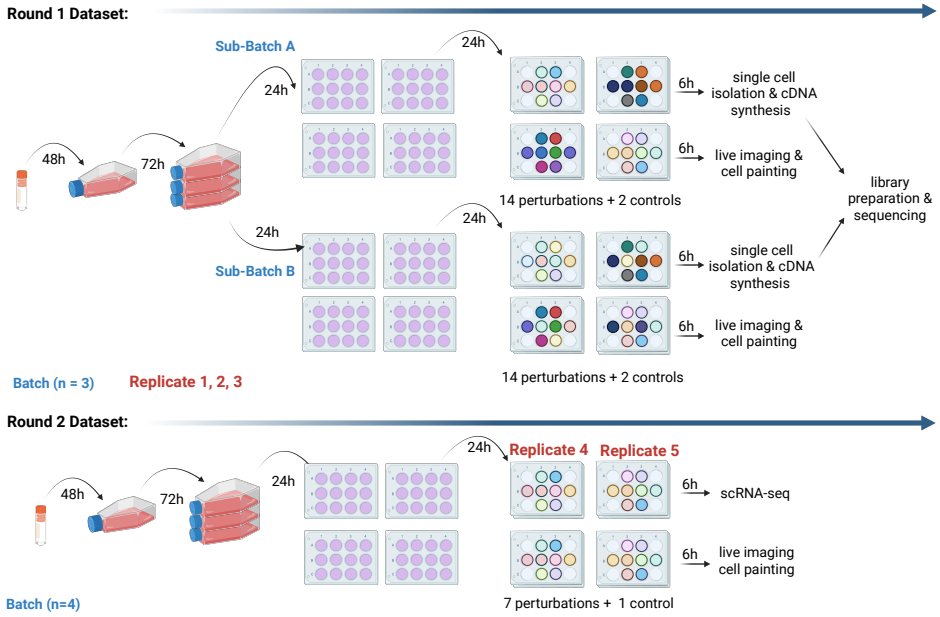

Figure 3: **Illustration of the sequential steps involved in experimental data generation rounds 1 and 2 resulting in the full scGeneScope dataset.** Multimodal datasets were created by pairing scRNA-seq with Cell Painting of U2-OS cells, which were perturbed by 28 chemicals and one solvent control (DMSO). 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.

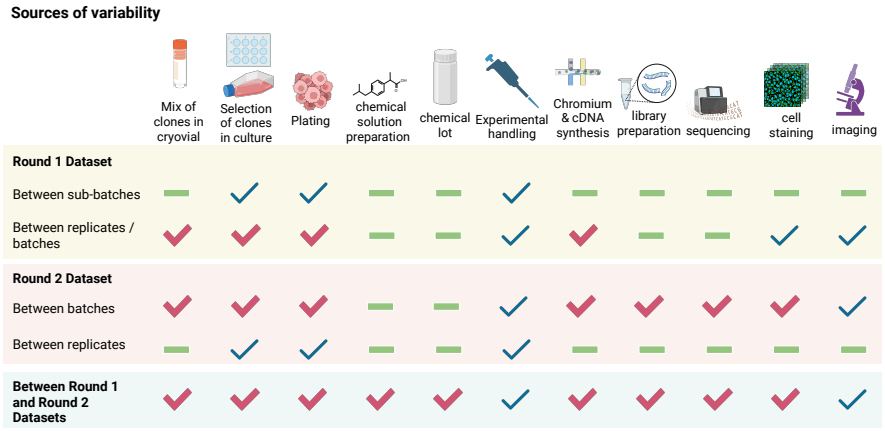

Figure 4: **Schematic representing sources of variability between sub-batches, batches, replicates, and datasets and their relative significance.** A bold red check mark indicates a significant source of variability, a thin blue check mark indicates a low-moderate source of variability, and a green minus 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.

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

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

A.1.2 Chemical Selection

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

We employed a two-step strategy to assemble a final panel of 28 small-molecule treatments used to create the scGeneScope dataset. We first leveraged a data-agnostic approach, where an existing U2-OS Cell Painting dataset was mined [40] for annotations that mapped compounds to their molecular targets. To maximize biological breadth, 14 non-overlapping targets were chosen, followed by nomination of one well-behaved chemical compound per target, prioritizing acceptable toxicity, solubility, assay robustness, and availability of appropriate controls.

Next, we expanded the panel by integrating data-driven insights from the Rosetta L1000 datasets [5] as well as from the CP-JUMP [40] library. The objective of the Rosetta L1000 analysis was to rank compounds across the LINCS and CDRP-Bio chemical perturbation datasets, which include screens against the U2-OS osteosarcoma cell line. Well-level L1000 measurements were downloaded, gene expression vectors were standardized and filtered, and the Pearson correlation coefficients between replicates were computed. Compounds with high replicate correlation were retained, and gene expression data was re-standardized. Compound effects were quantified via Euclidean distance between the gene expression vector for compound-treated cells and the gene expression vector for the on-plate control. Compounds were ranked by this distance metric to guide downstream nomination.

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

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

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

A.1.3 Biological Materials and Methods

Cell Culture U2-OS cells (ATCC #HTB-96) were cultured in McCoy's 5a Medium Modified (Gibco #16600-082) with 10% FBS (Omega #FB-01) in 37° C at 5% CO_2. Cells were plated in 12-well glass bottom plates (CellVis #P12-1.5H-N) 24 hours before the start of treatment in exact duplicate plates. Compounds were added to the wells in the doses listed in Table 2 in a final solution of 0.1% v/v dimethyl sulfoxide (Thermo Scientific Chemicals #J66650AP), with the location of compounds shuffled between consecutive rounds of replicates. Dosages were selected for each treatment by reviewing the dosages used in prior works [40, 2], and selecting ranges with low levels of toxicity. After 6 hours of treatment, plates were processed for scRNA-seq and Cell Painting.

Cell Painting After 5.5 hours of incubation with indicated treatments, Mitotracker Deep Red FM (CST #8778) was spiked into each well for a final concentration of 500 nM and incubated for 30 min. Cells were fixed in freshly diluted 3.2% v/v paraformaldehyde (EM Sciences #50980495) at room temperature for 20 min, washed three times with 1x HBSS (Gibco #14-175-103), and stained with Cell Painting staining mix (6 µM Syto14 (Fisher #S7576), 1.25 µL/mL Phalloidin AlexaFluor 568 (Fisher #A12380), 50 µg/mL Concanavalin A CF750 (Biotium #29080), 1.5 µg/mL WGA568 (Biotium #29077), 2 µg/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 µM
PQ401	IGF-1 / IGF-1R inhibitor	Cayman	196868-63-0	10 µM
Splitomicin	SIRT inhibitor	Cayman	138-433-9	22.22 µM
(R) -MG132	Proteasome inhibitor	Cayman	1211877-36-9	10.51 µM
(R) -Roscovitine	CDK inhibitor	Cayman	186692-46-6	14.11 µM
Wy 14643 / Pirinixic Acid	PPAR receptor agonist	Cayman	50892-23-4	10 µM
Fluocinonide	Glucocorticoid receptor agonist	Cayman	0356-12-7	4.04 µM
Caffeine	Adenosine receptor antagonist	Cayman	58-08-2	10 µM
LY303511 (hydrochloride)	Casein-kinase / mTOR / PI3K inhibitor	Cayman	854127-90-5	10 µM
Simvastatin	HMG-CoA-reductase inhibitor	Cayman	79902-63-9	10 µM
Colchicine	Microtubule inhibitor	Cayman	64-86-8	10 µM
Pantoprazole	ATPase inhibitor	Cayman	102625-70-7	10 µM
Benzbromarone	Chloride-channel blocker	Cayman	3562-84-3	4.72 µM
AMG-900	Aurora-kinase inhibitor	Cayman	945595-80-2	10 µM
DBeQ	p97 inhibitor	Cayman	177355-84-9	10 µM
Daporinad / FK-866	Transferase inhibitor	Cayman	658084-64-1	10 µM
Vorinostat / SAHA	Histone-deacetylase inhibitor	Cayman	149647-78-9	10 µM
Quinidine	Sodium-channel blocker	Cayman	56-54-2	10 µM
Aloxistatin / E-64d	Cysteine-protease inhibitor	Cayman	88321-09-09	10 µM
Cycloheximide	Protein-synthesis inhibitor	Cayman	66-81-9	10 µM
Thapsigargin	SERCA inhibitor	Cayman	67526-95-8	7.68 µM
BAY 11-7082	NF-κB-pathway inhibitor	Cayman	19542-67-7	24.14 µM
CGK-733	ATM/ATR-kinase inhibitor	Cayman	905973-89-9	10 µM
PD-98059	MEK / MAP-kinase inhibitor	Cayman	167869-21-8	18.71 µM
GW-843682X	PLK inhibitor	Cayman	660868-91-7	10 µM
PMA	PKC activator	Cayman	16561-29-8	3.06 µM
SKI-II	Sphingosine-kinase inhibitor	Cayman	312636-16-1	10 µM
HARMAN	Monoamine-oxidase inhibitor	Cayman	486-84-0	10 µM

(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%CO_2 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/µl 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), Concanavlin 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 [50]. 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 [51]. CellPose

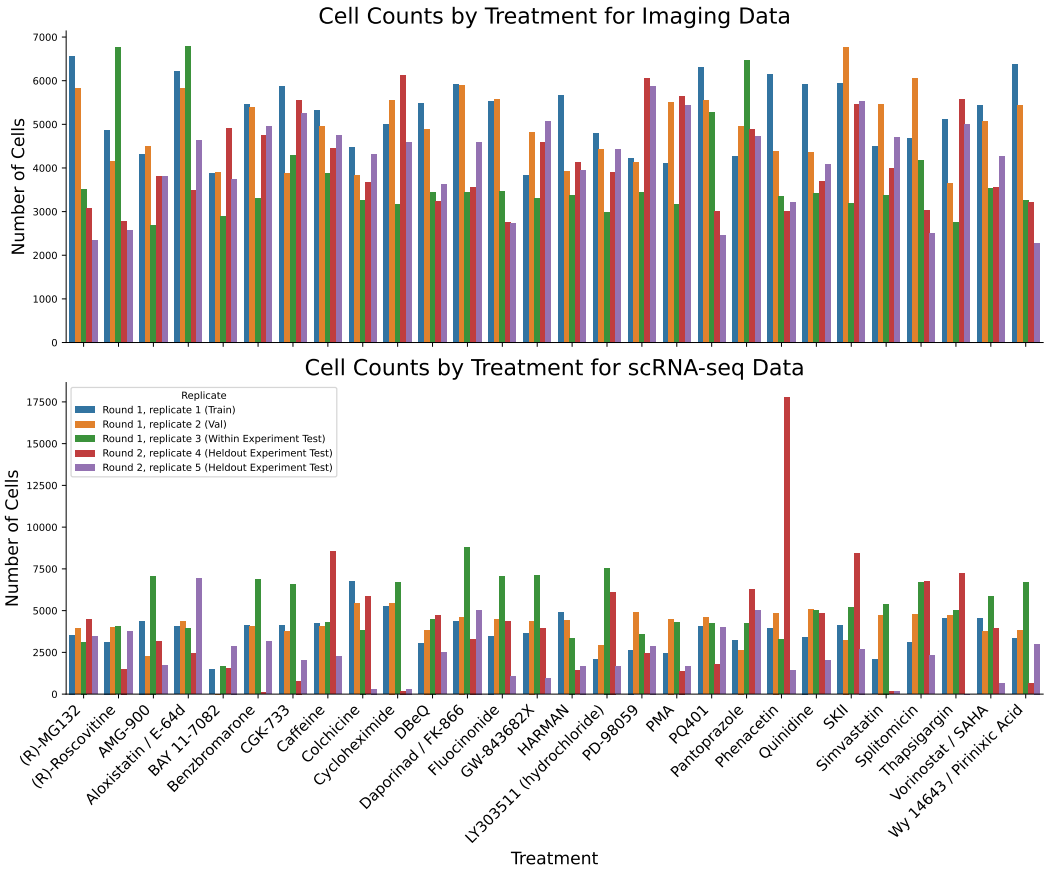

Figure 5: **Single-cell profile counts plotted by treatment and replicate for Cell Painting and scRNA-seq modalities.**

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 architectures

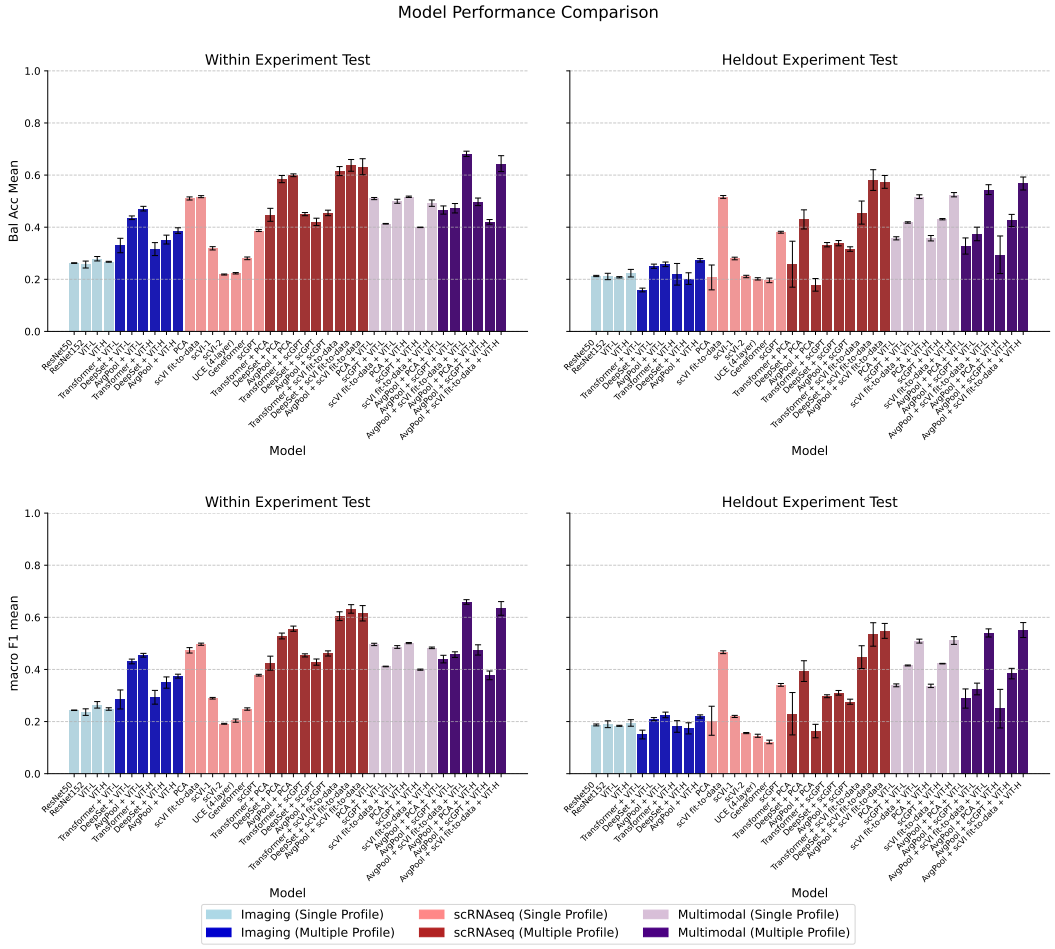

Figure 6: **Visualization of all results.** All results from Table 1 are visualized; the left column shows Within Experiment (WE) evaluations and the right column shows Heldout Experiment (HE) evaluations, while the top row shows Balanced Accuracy and the bottom row shows Macro F1 scores.

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-CLIP models were loaded with timm/resnet50_clip.openai pretrained weights, while ViT-H-CLIP models were loaded with CLIP-ViT-H-14-laion2B-s32B-b79K pretrained weights. OpenPhenom models were loaded with OpenPhenom-S/16 pretrained weights. ResNet50 and ResNet152 embeddings were of size 2048; ResNet50-CLIP, ViT-L-16, and ViT-H-CLIP embeddings were of size 1024; ViT-H-14 embeddings were of size 1280; OpenPhenom embeddings were of size 384. Embeddings were generated for each Cell Painting stain independently, triplicating each single channel stain image into the 3-channel inputs required for ImageNet-pretrained and CLIP-pretrained models. For OpenPhenom models, triplication of each channel was not required, but embeddings were similarly computed independently for each Cell Painting stain. Finally the embeddings for each stain were concatenated together together, giving total scImage embedding sizes of 10,240 for ResNet50 and ResNet152; 6,140 for ViT-H-14; and 5,120 for ViT-L-16, ResNet50-CLIP, and ViT-H-CLIP; and 1,920 for OpenPhenom.

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

Table 4: Multimodal single profile models hyperparameter ranges.

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

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

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

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

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

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

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

Table 8: Multimodal multi-profile models hyperparameter ranges.

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

A.2.3 Fitting Task Heads

The task heads are designed as a series of PyTorch operations mapping the high-dimensional feature space to the target output space. The architecture is defined within the model configuration, leveraging Hydra's configuration management capabilities to ensure flexibility and reproducibility. Data preprocessing is managed by a dedicated data module, implemented using PyTorch Lightning's `LightningDataModule`. This module is responsible for loading, augmenting, and batching the data, ensuring that it is in the appropriate format for the model. The data module's configuration is specified in `cfg.data` (see Section A.4.2 for further details), allowing for seamless integration with the training pipeline. The training of task heads is conducted using PyTorch Lightning's `Trainer` class. This process is augmented by a suite of callbacks that facilitate early stopping, learning 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.

A.4 Software Library

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

A.4.1 Frameworks

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

PyTorch: Serving as our core deep learning framework, `PyTorch` [52] provides efficient automatic differentiation and neural network primitives through a dynamic computational graph approach. We leverage `PyTorch`'s comprehensive ecosystem for building and optimizing neural architectures, with particular emphasis on its data handling abstractions. Specifically, we utilize the `torch.utils.data.Dataset` and `torch.utils.data.DataLoader` interfaces to implement our data processing pipeline in the `scgenescope.data` module, enabling efficient data loading and batching during model training.

PyTorch Lightning: To enhance code clarity and reduce boilerplate while maintaining flexibility, we employ PyTorch Lightning [53] as our high-level training framework. Lightning provides structured abstractions that separate model logic from training mechanics through its `LightningModule` and `LightningDataModule` interfaces. We encapsulate our models and datasets within these classes, allowing Lightning's `Trainer` to handle training loop logistics, device management, and distributed training seamlessly. Additionally, Lightning's callback system facilitates integration with monitoring tools like TensorBoard, enabling comprehensive experiment tracking.

Hydra: Complex machine learning pipelines require robust configuration management. We utilize Hydra [54] to provide a hierarchical configuration system that elegantly handles the complexity of our benchmark suite. Hydra enables composition-based configuration management, where different components can be configured independently and combined as needed. This approach facilitates reproducible experiments through comprehensive configuration logging, while providing convenient features such as command-line completion, hyperparameter optimization via Optuna, and configuration validation through type checking.

AnnData: For biological data management, we adopt AnnData [55] as our primary data structure. AnnData provides a specialized container for single-cell RNA sequencing data, storing both the gene expression matrix and associated metadata. Each AnnData object maintains cell-level annotations (including perturbation information and covariates) and gene-level metadata (such as gene identifiers). Our data module interfaces with AnnData through the h5ad file format, while our analysis module generates predictions in AnnData format, ensuring compatibility with the broader single-cell analysis ecosystem.

A.4.2 Data Abstractions

Our dataset presents unique challenges for data management and model training. Each treatment condition contains two large collections of single-cell profiles - one for scRNA-seq and one for scImage data. Due to memory constraints, training on complete cell populations is infeasible, necessitating 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.

```python
@dataclass
class DataStore(torch.utils.data.Dataset, Generic[T]):
    """A tabular single-cell dataset."""
    store: Indexed[T]
    observations: pd.DataFrame | None

@dataclass
class Samples(DataStore):
    """An interface for a single-cell profile dataset."""
    @classmethod
    def from_anndata(cls, ...):
        ...
    @classmethod
    def from_embeddings(cls, ...):
        ...
```

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

```python
@dataclass
class Population:
    """A dataset of single-cell profiles grouped by a condition."""
    samples: SupportsGrouping
    condition_on: str
    indexing: Index = Index.LOC
    transform: Callable | None = None
    include_condition: bool = False
    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.

```python
class ConditionAlignedPopulations(torch.utils.data.StackDataset):
    """A dataset of condition-aligned populations."""
    ...
    @classmethod
    def join_on_condition(
        cls,
        *dataset_factories: Callable[[], SupportsIndexModeSelect],
        condition: str,
        ...
    ) -> Self:
        ...
```

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`.

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

Listing 4: Sampled population class.

Data Pipeline Configuration: Each of the above classes is used in our data configuration system 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/
```

A.4.3 Reproducibility and Deterministic Datasets

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

A.4.4 Model Abstractions

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

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

- `encoder`: A neural network module that projects input data into a latent representation
- `classifier`: A module that maps the latent representation to treatment predictions
- `optimizer_factory`: A callable that constructs the optimizer for model parameters

- `scheduler_factory`: A callable that creates the learning rate scheduler
- `compile`: A boolean flag enabling PyTorch 2.0 compilation for improved performance

```python
class UnimodalSampleClassifier(LitClassifier):
    """A unimodal classifier.

    This model takes a single modality as input and outputs a prediction.

    Attributes:
        encoder: The encoder model.
        classifier: The classifier model.
    """

    def __init__(
        self,
        encoder: torch.nn.Module | None,
        classifier: torch.nn.Module,
        optimizer_factory: Callable[
            [Iterable[torch.nn.Parameter]], torch.optim.Optimizer
        ],
        scheduler_factory: Callable[[torch.optim.Optimizer], torch.optim.lr_scheduler],
        compile: bool,
    ) -> None:
```

Listing 5: Unimodal classifier implementation.

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

- `encoders`: A collection of modality-specific encoder networks, organized as either a `ModuleList` or `ModuleDict`
- `classifier`: A module that combines encoded representations to make predictions
- `loss`: The training objective function
- `normalize`: A flag indicating whether to normalize encoded representations before classification

```python
class MultiModalMultipleInputClassifier(LitClassifier):
    """A multi-modal classifier.

    This model takes multiple modalities as input and outputs a prediction.

    Attributes:
        encoders: The encoder models.
        classifier: The classifier model.
        loss: The loss function.
    """

    encoders: nn.ModuleList | nn.ModuleDict
    classifier: nn.Module
    loss: nn.Module
    normalize: bool
```

Listing 6: Multi-modal classifier implementation.

A.4.5 Experiment Configuration

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

```
experiment/
```

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

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

```
# Run single-profile imaging experiment with ViT-H
train experiment=imaging/singleprofile/train_on_concat_imagenet_vit_h.yaml

# Run multi-profile RNA-seq experiment with PCA embeddings
train experiment=rnaseq/multiprofile/train_avgpool_on_pca_n2000.yaml

# Run multi-modal experiment combining ScGPT and ViT-L
train experiment=multimodal/multiprofile/train_avgpool_on_scgpt_with_concat_imagenet_vit_l.yaml
```

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

```
# Explore extra parameters to override in src/scgenescope/config
train experiment=<config_path>
    # Override trainer parameters
    trainer.max_epochs=100 trainer.accelerator=gpu trainer.devices=[0]
    # Override model parameters
    model.optimizer_factory.lr=1e-4
```

Imaging Configurations: The imaging modality configurations support both single-profile and 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_h_clip.yaml
├── train_on_concat_imagenet_vit_l.yaml
├── train_on_concat_openphenom.yaml
├── train_on_concat_resnet152.yaml
├── train_on_concat_resnet50.yaml
└── train_on_concat_resnet50_clip.yaml
```

Multi-modal Configurations: The multi-modal configurations integrate RNA-seq and imaging 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
```

RNA-seq Configurations: The RNA-seq configurations support various embedding approaches 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
```

A.5 Class-wise Performance and Confusion Matrix Analysis

In this section we present a confusion matrix analysis of the best unimodal single profile models (scVI fit-to-data and ViT-L), as well as their combination in the multimodal single profile model. This analysis reveals that single cell and imaging models have distinct and often complementary error patterns when classifying treatments (Supplemental Figure 7). Although scVI outperforms ViT-L on average, there are treatments that imaging can classify strongly but not scRNA-seq. For instance, for the treatment Aloxistatin/E-64d, imaging achieves a 77.9% true positive rate (TPR) in WE and 91% TPR in HE, but scRNA-seq achieves 8.2% TPR in WE and 6.9% in HE (Supplemental Figure 8). Intriguingly, we found cases where multimodal integration yielded higher performance than both unimodal models, with the effect more pronounced in the HE evaluation, indicating stronger generalization. For instance, with the treatment PD-98059, scVI achieves 37.2% TPR, ViT only 0.42%, but the multimodal model achieves 66% TPR in HE. Unfortunately, there are also a number of cases, such as the treatment HARMON, where the performance drops from 27.35% for scRNA-seq to 1.7% in multimodal case, even in the WE test scenario. Overall, this analysis drives additional insight that the imaging and transcriptomic modalities have complementary signal, and that it is possible for multimodal integration to produce synergistic effects between modalities, but that additional work is needed to solidify these gains.

A.6 Existing Dataset Reference

In Table 9, we provide a reference table of existing datasets. scGeneScope is the only dataset we are aware of with perturbationally paired scRNA-seq and single cell Cell Painting images.

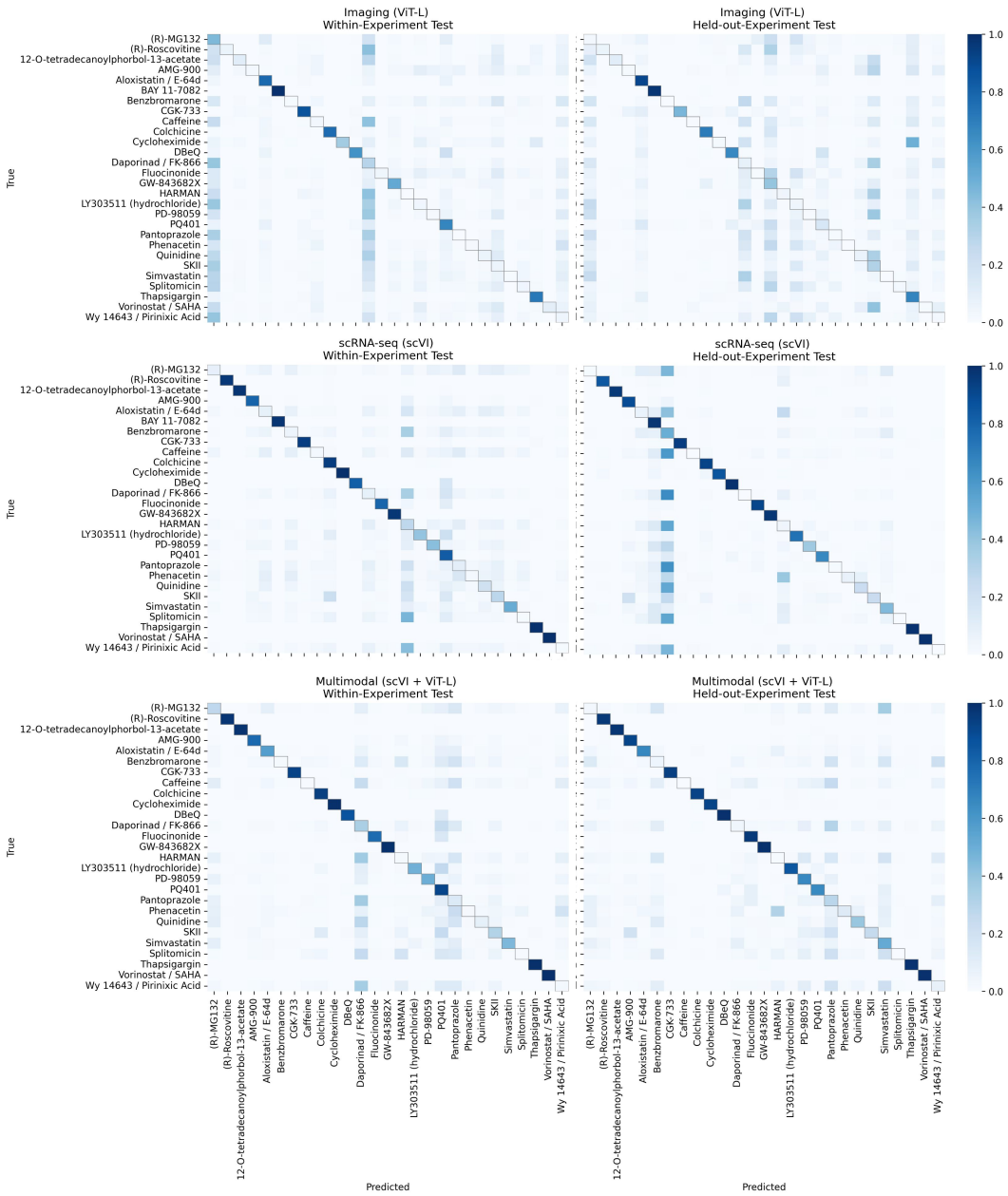

Figure 7: **Confusion matrix** analyses for best performing imaging (ViT-L), transcriptomic (scVI), and multimodal (AvgPool scVI + ViT-L) models reveal unique error patterns across modalities and generalization conditions.

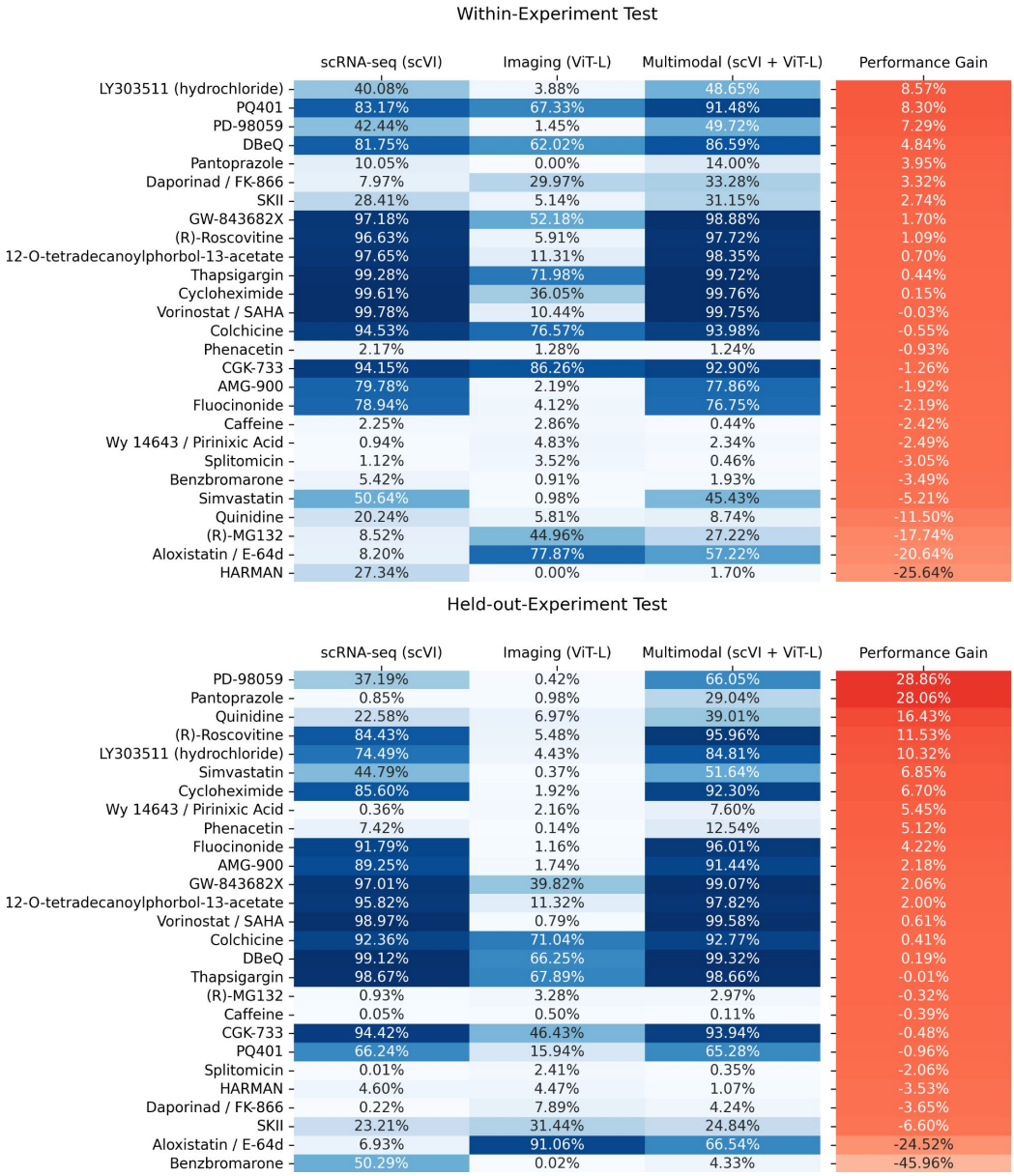

Figure 8: **Classwise true positive rates** (TPR) for best performing imaging (ViT-L), transcriptomic (scVI), and multimodal (scVI + ViT-L) models ranked by change in performance in the multimodal model reveal cases where multimodal integration helps or hurts classification accuracy, with the change computed as the multimodal TPR minus the better of the two unimodal TPRs.

Dataset	Cell Line(s)	# Perturbations	# Cells / Samples	Benchmark Task	Modality	Citation / Link
LINCS	77	19,811 small molecule compounds	∼1.3M profiles	Drug response, gene expression	L1000	Subramanian et al. 2017
SciPlex	3 (A549, K562, MCF7)	188 compounds	∼5,000 samples	Drug / dose response	scRNA-seq	Srivatsan et al. 2019
Tahoe	50	379 compounds, 52,886 cell line–drug–dose conditions	100.6M cells	Drug response	scRNA-seq	Zhang et al. 2025
scPerturb	44 datasets (various cell lines, primary cells, tissues)	∼1M (genetic + chemical)	∼5M cells	Drug response	scRNA-seq	Peidli et al. 2024
Cell Painting	13 cell cultures (lines, primary, coculture)	∼1K for large screens	17,280 / 384-well plate	Morphological profiling	Microscopy images	Bray et al. 2016
JUMP CP	1 (U2OS)	∼136,000 (compounds + genes)	1.6B cells	Morphological profiling	Microscopy images	Chandrasekaran et al. 2023
RxRx3-core	1 (HUVEC)	1,674 compounds, 736 CRISPR KOs	∼200K images	Morphological profiling	Microscopy images	Kraus et al. 2025
BBBC036/022/037	1 (U2OS)	30K / 1.6K / 323 ORF constructs	∼5M / 40K / 100K	Morphological profiling	Microscopy images	Ljosa et al. 2012

Table 9: Overview of the existing data landscape for unimodal perturbation datasets in transcriptomics and imaging modalities.

Table 10: ViT-L (ImageNet), AvgPool multiprofile: k-scan on WE/HE. Values are mean ± std.

	Within-Experiment Test		**Held-out-Experiment Test**	
k	**Bal. Acc.**	**Macro-F1**	**Bal. Acc.**	**Macro-F1**
2	0.3121 ± 0.0040	0.2941 ± 0.0050	0.2087 ± 0.0054	0.1877 ± 0.0045
4	0.3425 ± 0.0027	0.3245 ± 0.0043	0.2250 ± 0.0054	0.1985 ± 0.0071
8	0.3772 ± 0.0201	0.3607 ± 0.0217	0.2450 ± 0.0082	0.2135 ± 0.0089
16	0.4376 ± 0.0130	0.4221 ± 0.0126	0.2645 ± 0.0075	0.2299 ± 0.0086
32	0.4695 ± 0.0117	0.4515 ± 0.0062	0.2672 ± 0.0068	0.2367 ± 0.0078
64	0.4960 ± 0.0086	0.4805 ± 0.0034	0.2574 ± 0.0048	0.2261 ± 0.0081
128	0.4955 ± 0.0092	0.4739 ± 0.0098	0.2504 ± 0.0045	0.2190 ± 0.0056
256	0.4919 ± 0.0040	0.4540 ± 0.0060	0.2339 ± 0.0157	0.2079 ± 0.0170

Table 11: scVI (n=200), AvgPool multiprofile: k-scan on WE/HE. Values are mean ± std.

	Within-Experiment Test		**Held-out-Experiment Test**	
k	**Bal. Acc.**	**Macro-F1**	**Bal. Acc.**	**Macro-F1**
2	0.5611 ± 0.0066	0.5517 ± 0.0046	0.5473 ± 0.0042	0.5353 ± 0.0052
4	0.5948 ± 0.0107	0.5867 ± 0.0089	0.5683 ± 0.0064	0.5582 ± 0.0092
8	0.6109 ± 0.0092	0.6023 ± 0.0144	0.5488 ± 0.0133	0.5402 ± 0.0204
16	0.6262 ± 0.0167	0.6143 ± 0.0167	0.5544 ± 0.0056	0.5405 ± 0.0155
32	0.6321 ± 0.0307	0.6156 ± 0.0295	0.5739 ± 0.0246	0.5480 ± 0.0286
64	0.6299 ± 0.0173	0.6141 ± 0.0155	0.5504 ± 0.0187	0.5246 ± 0.0215
128	0.6275 ± 0.0073	0.5991 ± 0.0102	0.5373 ± 0.0169	0.5056 ± 0.0171
256	0.6375 ± 0.0117	0.6005 ± 0.0167	0.5466 ± 0.0192	0.5145 ± 0.0235

A.7 Tuning the cardinality of multiprofile aggregation

As discussed in Section 3.3, we explored the performance of the unimodal and multimodal models with multiple numbers of profiles per modality. The results shown in Table 1 are using $k = 32$ profiles per modality. We varied the number of profiles per modality from 2 to 256, and evaluated the performance of the models on the test sets. We found that the performance of the models generally improved as the number of profiles per modality increased, but that the improvement was not always consistent especially on the HE data. See Table 10 and Table 11 for the results of varying the k.

B scGeneScope Datasheet

B.1 Motivation

B.1.1 For what purpose was the dataset created?

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

B.1.2 Who created the dataset?

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

B.1.3 Who funded the creation of the dataset?

Altos Labs funded the creation of the scGeneScope dataset.

B.1.4 Any other comments?

No.

B.2 Composition

B.2.1 What do the instances represent?

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

B.2.2 How many instances are there in total?

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

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

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

B.2.4 What data does each instance consist of?

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

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

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

B.2.6 Is any information missing from individual instances?

No information is withheld from the individual instances.

B.2.7 Are relationships between instances made explicit?

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

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

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

B.2.9 Errors, noise, or redundancies?

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

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

The dataset is self contained within two h5ad files containing the scRNA-seq data (one for round one and one for round two) and two folder systems storing the Cell Painting imaging data. The data is hosted at https://huggingface.co/datasets/altoslabs/scGeneScope, with tutorials on how to access and use the data available at our github.

B.2.11 Confidential data?

There is no confidential data in the scGeneScope dataset.

B.2.12 Potentially offensive or sensitive content?

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

B.2.13 Does the dataset relate to people?

The scGeneScope dataset does not relate to people.

B.2.14 Identification of sub-populations?

The scGeneScope does not identify any sub-populations.

B.2.15 Possibility of identifying individuals?

The scGeneScope does not identify any individuals.

B.2.16 Sensitive attributes present?

There are no sensitive attributes in the scGeneScope dataset.

B.2.17 Any other comments?

No.

B.3 Collection Process

B.3.1 How was the data acquired?

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

B.3.2 Mechanisms or procedures used?

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

B.3.3 Sampling strategy (if any)?

All samples which passed quality control where used.

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

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

B.3.5 Timeframe of data collection?

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

B.3.6 Ethical review processes?

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

B.3.7 If people are involved:

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

B.3.8 Any other comments?

No.

B.4 Pre-processing / Cleaning / Labeling

B.4.1 Was any pre-processing done?

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

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

At the time of paper submission, the raw data is saved but not yet released alongside the preprocessed data for technical and storage related reasons. We release both the imaging and scRNAseq data in preprocessed forms commonly used by the field, and can release the raw data if needed.

B.4.3 Is the preprocessing software available?

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

B.4.4 Any other comments?

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

B.5 Uses

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

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

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

This is the first paper that has used this dataset.

B.5.3 Other potential tasks?

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

B.5.4 Factors that might impact future uses?

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

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

B.5.6 Any other comments?

No.

B.6 Distribution

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

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

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

The scGeneScope dataset is hosted at `https://huggingface.co/datasets/altoslabs/scGeneScope`, with download and usage instructions in our code.

B.6.3 When will the dataset be distributed?

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

B.6.4 License or terms of use?

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

B.6.5 Third-party IP or other restrictions?

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

B.6.6 Export controls or regulatory restrictions?

The scGeneScope has no export controls or regulatory restrictions.

B.6.7 Any other comments?

No.

B.7 Maintenance

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

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

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

The corresponding authors of the paper can be contacted for questions regarding the data.

B.7.3 Erratum available?

B.7.4 Will the dataset be updated?

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

B.7.5 Retention limits for data relating to people?

The data has no retention limits or relation to people.

B.7.6 Will older versions remain available?

The current version will remain available.

B.7.7 Mechanism for external contributions / extensions?

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

B.7.8 Any other comments?

No.

NeurIPS Paper Checklist

1. **Claims**

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

   Answer: [Yes]

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

   Guidelines:

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

2. **Limitations**

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

   Answer: [Yes]

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

   Guidelines:

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

3. **Theory assumptions and proofs**

   Question: For each theoretical result, does the paper provide the full set of assumptions and 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**

Question: Does the paper provide open access to the data and code, with sufficient instructions to faithfully reproduce the main experimental results, as described in supplemental material?

Answer: [Yes]

Justification: We release our dataset via HuggingFace. We release our code via github, with sufficient instructions to reproduce our main results.

Guidelines:

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

6. **Experimental setting/details**

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

Answer: [Yes]

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

Guidelines:

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

7. **Experiment statistical significance**

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

Answer: [Yes]

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

Guidelines:

- The answer NA means that the paper does not include experiments.
- The authors should answer "Yes" if the results are accompanied by error bars, confidence intervals, or statistical significance tests, at least for the experiments that support 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 alongise 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.