

Foundation Models Improve Perturbation Response Prediction

Elijah Cole^{*†}, Geert-Jan Huizing[†], Sohan Addagudi[†],
Nicholas Ho[†], Euxhen Hasanaj[†], Merel Kuijs[†], Toby Johnstone[†],
Maria Carilli, Alec Davi, Caleb Ellington, Christoph Feinauer,
Pan Li, Romain Menegaux, Shahin Mohammadi, Yanjun Shao,
Josiah Zhang, Emma Lundberg[§], Le Song, Ziv Bar-Joseph^{*},
Eric P. Xing^{*}

GenBio AI, Palo Alto, CA, USA.

^{*}Corresponding author(s). E-mail(s): elijah.cole@genbio.ai,
ziv.bar-joseph@genbio.ai, eric.xing@genbio.ai.

[†]Core Contributor

Abstract

Predicting cellular responses to genetic or chemical perturbations has been a long-standing goal in biology. Recent applications of foundation models to this task have yielded contradictory results regarding their superiority over simple baselines. We conducted an extensive analysis of over 600 different models across various prediction tasks and evaluation metrics, demonstrating that while some foundation models fail to outperform simple baselines, others significantly improve predictions for both genetic and chemical perturbations. Furthermore, we developed and evaluated methods for integrating multiple foundation models for perturbation prediction. Our results show that with sufficient data, these models approach fundamental performance limits, confirming that foundation models can improve cellular response simulations.

Code and Data:

<https://github.com/genbio-ai/foundation-models-perturbation>

[§]Department of Bioengineering, Stanford University, CA, USA.

1 Introduction

Predicting the outcome of cellular perturbations is a long standing challenge in molecular biology [1–3]. Typically these perturbations are either genetic modifications (e.g. knocking out or over-expressing one or more genes [4]) or chemical treatments (e.g. exposing cells to a small molecule [5]). Readout types for perturbation experiments include cell death [6, 7], cell growth [8], and imaging [9]. One of the most interesting readouts is gene expression [10], which measures the impact of a perturbation on some or all genes in a cell. Initial work focused on bulk expression [11] while more recent analysis considers single cell perturbation responses [12].

There are several reasons why perturbation prediction is an important question in biology. First, the ability of models to predict the impact of a perturbation in a specific context (cell type, tissue, disease state, etc.) suggests that such models may have learned something useful about the underlying molecular networks that are active and repressed in that context [13]. A model that accurately predicts perturbation responses in diverse settings may have learned to account for both the activity of the knocked out gene (or the target of the small / large molecule) and the interactions between that gene / protein and other proteins and pathways in the cell. Such knowledge is one of the main goals of systems biology. In addition to demonstrating that a useful representation of cell state and interactions has been learned, such analysis is also a key part of most drug discovery and design processes. The ability to accurately predict the impact of a specific treatment at the cellular level can save both time and work when identifying new interventions for diseases [14]. In addition, such models can provide information not just on the direct effects of a treatment but also alert researchers to potential indirect or side effects, which in many cases are only discovered very late in the drug discovery process [15]. Accurate perturbation response models would be impactful for both basic and applied biology research, leading to increased interest in such models.

Early work on perturbation modeling aimed to predict drug mechanism of action [16], estimate sensitivity to drugs [17], or model gene regulatory networks with differential equations [1, 18, 19]. As the available data grew, perturbation response prediction problems were studied using statistical learning methods including Elastic-Net [6, 20, 21], neural networks [14], and matrix factorization models [22, 23]. Early applications of deep learning include Dr.VAE [24] and scGen [25]. These initial deep learning frameworks have been extended in several directions. CPA [26] enabled the prediction of combinatorial perturbation effects. GEARS [27] incorporated knowledge graphs and allowed for the prediction of perturbations not seen during training. Most recently, transformer-based architectures have been adapted from the mainstream ML community, leading to models like Geneformer [28], scFoundation [29], and scGPT [30], which pretrain on massive single-cell atlases and provide schemes to use the FM for perturbation response prediction. These recent papers have claimed to achieve strong results using such methods for perturbation analysis. However, other recent publications [31, 32] take a contradictory view, arguing that FMs do not improve on simple baseline methods.

Here we perform a comprehensive study to resolve this issue by evaluating the use of biological FMs for perturbation prediction. We evaluated hundreds of different

embedding variants from frozen and fine-tuned FMs and compared them to simple baselines, classical ML methods, and recently published state-of-the-art AI methods across several prediction tasks for different perturbation types. Inspired by practices in mainstream machine learning, we take an embedding-centric view. We generate perturbation embeddings using FMs trained for modalities like DNA sequences [33], protein sequences [34], protein structures [35], and small molecules [36]. Our analysis indicates that while some FMs indeed perform no better than simple baselines, others (particularly interactome-based FMs) significantly outperform simple baselines across multiple tasks and cell types. We further show that integrating FMs using attention-based fusion leads to the best results for genetic perturbation prediction, in some cases greatly outperforming standard ML based methods.

2 Results

2.1 Embeddings vary dramatically in utility for genetic perturbation prediction

Some prior studies have claimed that there is no clear benefit from using FMs for predicting the effect of unseen perturbations, arguing that simple baselines (e.g. linear models with embeddings derived from PCA) perform as well or better (e.g. [31, 32, 37]). We observe similar trends when benchmarking various FM embeddings on the popular dataset from Norman et al. [4] (Figure S1).

However, larger datasets like Essential (4 cell lines, ~2000 perturbations each) reveal robust and reproducible differences between embeddings (Figure 1, Figure 2). While some embeddings indeed perform no better than simple baselines, other embeddings are much better. Interestingly, the performance of different embeddings seems to be driven primarily by the modality from which they are derived. This is consistent with findings in Littman et al. [38]. This observation is reinforced by Figure 3, which shows that the embedding sources themselves tend to cluster by modality even if the modeling details differ significantly (Supplemental Methods). This is likely due to some combination of the information in the modality itself and the model classes that tend to be used for each modality (e.g. transformers are common for scRNA-seq data, while graph neural networks are more natural for network data).

Figure 2 shows that the strongest embeddings are those derived from prior knowledge like interactome data (e.g. WaveGC), functional annotations of genes (GenotypeVAE), or textual descriptions of genes (e.g. GenePT). These three types significantly outperform simple baselines like PCA computed on the control cells. Textual descriptions can be viewed as a “soft” form of interactome data, where the relationships between entities are described in text instead of encoded as edges in a graph. A similar claim can be made for gene function annotations. If we accept this view, then we can conclude that interactome data is the best single source of embeddings for perturbation response prediction. Performance trends are similar if we use lasso instead of kNN regression (Figure S2). In addition, we consider an alternative (discrete) problem formulation based on differentially expressed gene (DEG) prediction (Figure S4). See Supplemental Methods for details.

Biological foundation models for scRNA-seq, DNA sequences, protein sequences, and protein structure all underperform the interactome-based embeddings. Among these foundation models, STRING sequence and some expression-based models (scGPT [30], AIDO.Cell [39]) perform best, and clearly outperform PCA. We note that larger models (for example, AIDO.cell models with more parameters) perform better, suggesting that model scaling may be useful for learning biologically relevant gene embeddings.

The results vary by cell line, but some are surprisingly strong. For example, on the K562 cell line the best embedding closes 77% of the gap between the train mean and the estimated experimental error limit (Methods). These results are especially impressive given that the predictive model is a simple kNN regression. The critical factor seems to be the choice of FM.

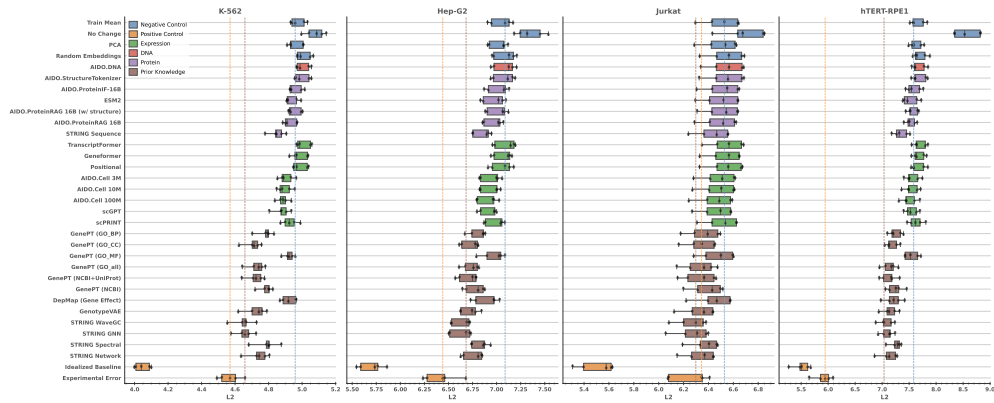


Fig. 1 Log fold change (LFC) regression results on the Essential dataset. Some embeddings consistently perform better than others, and this seems to be primarily explained by the underlying data modality. All results use kNN regression except for some baselines (Train Mean, No Change, Experimental Error). We perform 5-fold cross-validation for each method, and each data point corresponds to one fold. See Supplemental Results for metrics for over 600 models.

In total we tested more than 600 different models for this analysis and in general we observe the same trend as seen in Figure 1.

2.2 Fine-tuning improves performance for some models

So far the analysis has focused on directly using embeddings from FMs (Methods). While FMs can learn powerful representations of biology through large-scale pretraining, fine-tuning may further improve their performance [28, 30, 40]. To evaluate this in the context of perturbation modeling, we fine-tuned two of the best performing models from different categories (AIDO.Cell (3M) and STRING GNN) on the LFC regression task on Essential (Figure 4).

Our results indicate that the success of fine-tuning is highly dependent on both the architecture and the adaptation method employed. For AIDO.Cell (3M), the In-Silico KO method (Methods) provided a significant performance boost, outperforming both the kNN baseline and a nonlinear MLP ablation. This suggests that the model is capturing task-relevant biological signals beyond simple embedding proximity. Conversely, the Indexing method for AIDO.Cell and our fine-tuning attempts for STRING GNN resulted in performance degradation relative to frozen embeddings.

These discrepancies highlight a broader challenge in the field: the relatively small scale of current perturbation datasets makes foundation models highly susceptible to overfitting during fine-tuning. Leveraging fixed FM embeddings may currently be more appropriate and robust for downstream biological modeling.

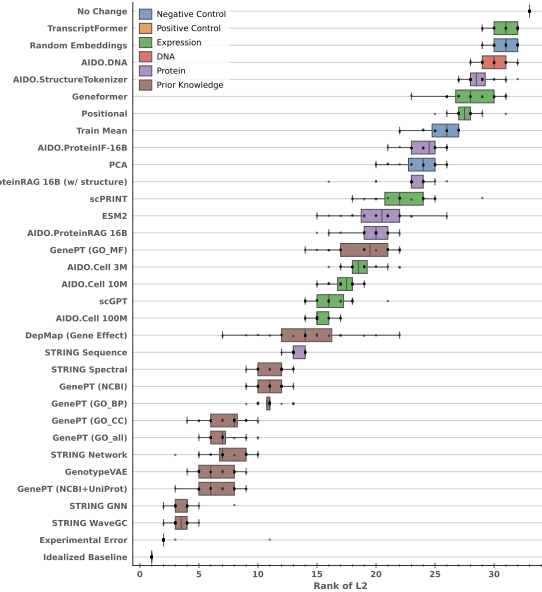


Fig. 2 Distribution of rankings for each embedding over 20 trained kNN regression models (4 cell lines, 5-fold cross-validation) for LFC regression. Note that the top 10 performing embeddings are all derived from prior knowledge.

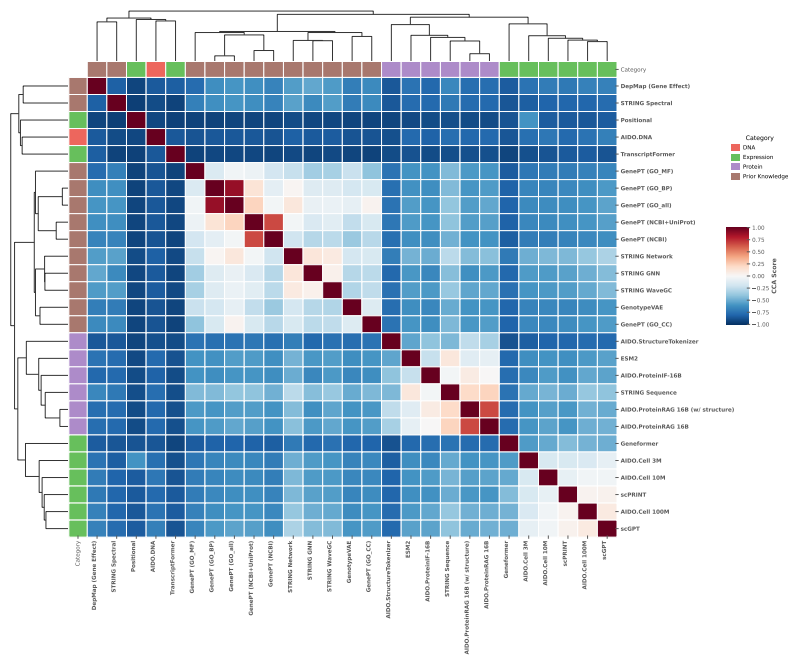


Fig. 3 Clustermap of embedding source similarity computed via CCA. Embeddings derived from the same types of information tend to be more similar, even if the models differ significantly.

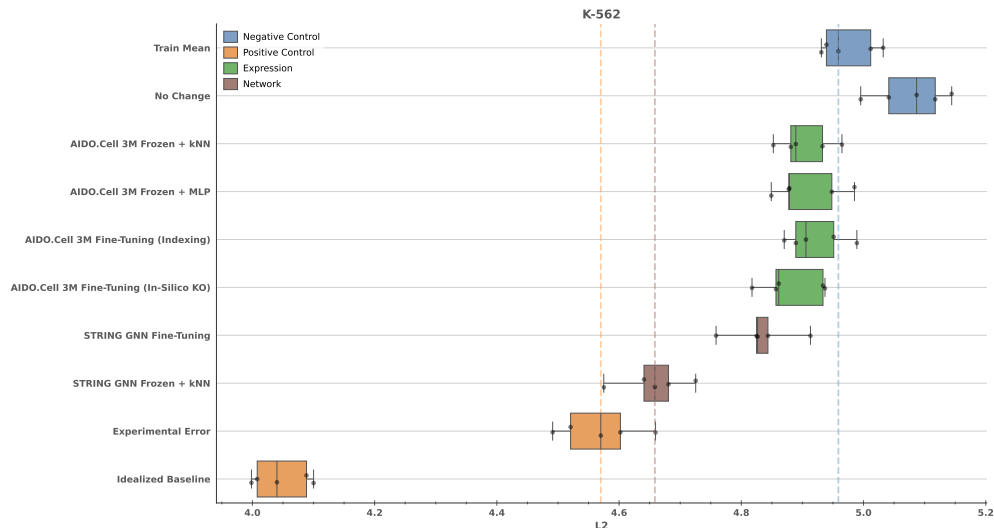


Fig. 4 LFC regression results on the K562 cell line in Essential. For AIDO.Cell (3M), one fine-tuning method (in-Silico KO) improves performance and one fine-tuning method (Indexing) hurts performance. As an ablation, we also consider an MLP which performs better than kNN but worse than In-Silico KO fine-tuning. For STRING GNN, fine-tuning significantly degrades performance. Methods are sorted by median L2 error over 5 folds. Train mean and experimental error are included for reference.

2.3 Complex embedding translation methods do not significantly outperform simple ones

The benchmarking results presented in Figure 1 use kNN for translating embeddings to average expression change vectors (Methods). However, there have been several suggestions in the literature for a different formulation based on modeling populations of single cells [41–43].

To test these, we used our Essential benchmark to evaluate several distribution prediction methods (Latent Diffusion [44], Flow Matching [45], Schrödinger Bridge [46]), which predict distributions of perturbed cells given a perturbation embedding and a collection of control cells. Note that there are specific applications of these ideas in the context of perturbation modeling [41–43]. We do not consider these algorithms, and instead study simple implementations of Latent Diffusion, Flow Matching, and Schrödinger Bridge. This allows us to decouple the embeddings used for perturbation conditioning from the generative model itself, enabling clear comparisons to the other methods we consider in this work.

We evaluate these methods in the LFC regression formulation by reducing their predicted distributions to their means; full details can be found in Supplemental Methods. In addition, we evaluated a published GNN-based perturbation model (GEARS). As can be seen in Figure S14, none of these advanced methods outperform kNN combined with our strongest embedding. This is especially notable for Latent Diffusion, Flow

Matching, and Schrödinger Bridge since those methods start from the same embedding as kNN. These methods are significantly more computationally expensive and difficult to tune (Methods). Our results suggest such methods currently they do not provide much practical utility for the formulation of perturbation response prediction we study here.

This work is primarily concerned with predicting the effects of unseen perturbations in seen cell lines. Recent works like STATE [47] consider a different context: predicting seen perturbations in cell lines for which limited data (few shot) or no data (zero shot) is available. We thus also compared an embedding-based MLP approach against STATE on the Essential (Figure S12) and Tahoe (Figure S13) datasets. The embedding-based MLP performs similar to or better than STATE in 3 out of 6 dataset-metric combinations we evaluate, improving on STATE by up to 64%. However, it does not seem that the embeddings themselves are the primary reason for this performance gain. In an ablation where we replace the embedding with a one-hot encoding of the perturbation, the performance of our MLP does not change much in either dataset for most metrics.

2.4 FMs can also improve small molecule perturbations predictions

For genetic perturbation datasets like Essential, perturbations can be naturally represented by embeddings of genes or proteins. The picture is more complex for chemical perturbation datasets like Tahoe (Figure S15). The traditional way to embed a small molecule is to use the SMILES or graph representation of the molecule to produce an embedding, either with a traditional feature encoding (e.g. Morgan fingerprints [48, 49]) or a transformer-based encoder (e.g. ChemBERTa [36]). An alternative is to represent a molecule by embedding its target (either the gene or a protein product) with a foundation model. However, this requires additional information that is not always available. A third approach is to take textual descriptions of the molecule from a database (e.g. ChEMBL) and use an LLM to reduce that text to a single embedding. Again, the utility of this depends on what information is available in databases.

We performed a comprehensive study comparing several traditional and FM based methods using the Tahoe-100M dataset. These were evaluated in a benchmarking setting similar to our Essential results. As before, the task is to predict the effect of unseen perturbations. In the LFC regression formulation, we find that no embedding convincingly outperforms the negative controls (Figure S5, Figure S6).

However, the DEG formulation of Tahoe shows that some embeddings clearly outperform negative controls (Figure S7). As a category, target-based embeddings performed best - simple target encodings, AIDO.Cell 100M, and scPRINT all performed well. The best-performing molecular structure fingerprint (ECPF:2) fingerprint was not far behind. The SMILES-based foundation models (with the exception of Mini-Mol) tended to underperform, possibly due to the fact that they have mostly been developed with chemical (not biological) applications in mind.

We repeated this analysis with the Sciplex-3 dataset. As in Tahoe, the signal in the LFC regression formulation was minimal (Figure S8). However, the DEG formulation

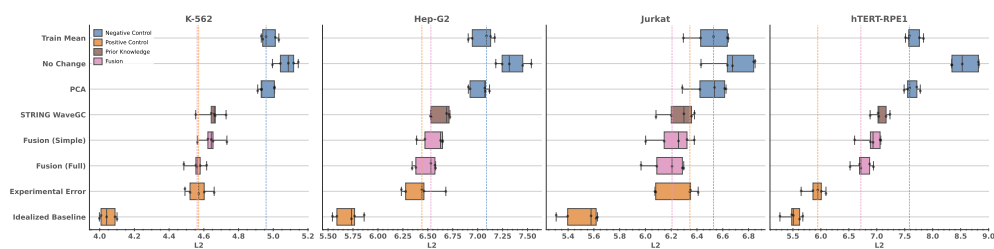


Fig. 5 LFC regression results for Essential. We compare the best unimodal embedding (WaveGC) against the simple fusion model and the full fusion model. We include negative controls and positive controls for context. Note that the fusion models are integrating all unimodal embeddings shown in Figure 1.

pared only slightly better (Figure S10). This may be because Sciplex has far fewer conditions for training than Tahoe.

2.5 Integrating embeddings from diverse FMs can further improve performance

All of the results presented so far have used a single perturbation embedding at a time to predict perturbation effects. However, objects like small molecules, genes, and proteins are complex and multifaceted. Any given representation may capture some aspects and neglect others. It therefore makes sense to integrate information from different models and modalities. In this section, we introduce and evaluate an attention-based model for integrating embeddings and predicting perturbation response (Methods).

Our results for Essential in Figure 5 show that multimodal fusion always improves over the best unimodal embedding, and that the more complex Fusion (Full) model improves over the Fusion (Simple) model. In fact, for two cell lines (Jurkat, K-562) the fusion result matches the estimated experimental error, which is the best achievable performance. Fusion results for the other two cell lines are also better than a single embedding, bridging 86% (Hep-G2) and 53% (hTERT-RPE1) of the gap between random and the experimental error bound. Our embedding fusion method achieves nearly maximal performance on Essential, so it may be necessary to create more difficult train/test splits in future work. For the DEG formulation of Essential, we do not see consistent gains for fusion (Figure S4).

For Sciplex and Tahoe, the LFC formulation seemed to show minimal signal so we did not attempt to train fusion models in those contexts. However, we did train simple fusion models for the DEG formulations (Figure S10, Figure S7). The fusion performance was no better than random in this context.

3 Discussion

The use of FMs for perturbation prediction has received a lot of attention over the last few years with contradictory findings. Here we performed a comprehensive analysis of embedding-based perturbation response prediction for both genetic and small molecule

interventions. As we showed, while some FMs fail to improve over simpler methods for this task, others provide significant gains. Interactome based FMs, which are based on the underlying networks activated in the cell, performed particularly well.

We also observe that predicting genetic perturbations is an easier task than predicting the impact of a small molecule intervention. This is true even when the target is (assumed to be) known for the small molecule. This is may be due to the higher specificity of genetic perturbations (i.e. small molecules are more likely to target multiple proteins or pathways). In addition, the search space of possible small molecules is much larger (estimated to be $\sim 10^{60}$ [50]) than the one we use for genes (which is limited to $\sim 2 \times 10^4$). Further, we usually have much less information on the underlying interaction networks for small molecules which, as we showed, provide the most valuable information for perturbation prediction.

While we observed that specific FM types outperform other types, integrating several different types of FMs provided the most accurate predictions for genetic perturbation. This is consistent with prior works that have experimented with combining two embeddings at a time [51, 52]. We have tested various strategies for integrating many embeddings and shown that in some cases, embedding fusion is sufficient to reach the estimated experimental error limit. In the small molecule setting, fusion of FMs did not improve performance. We speculate that this results from the weak performance of most individual embeddings and the potential to overfit when learning the more complex fusion model with a smaller number perturbation-context pairs.

Our study provides strong support for the use of FMs for perturbation prediction. Still, there are many open questions and potential research directions to improve and follow up on the results we presented. One such direction relates to the type of data that is most useful for perturbation prediction. As we showed, for genetic perturbations interactome data seems to be the most valuable. This should be taken into account by organizations that are aiming to study and develop virtual cell models. Specifically, this result indicates that data related to interactions in different contexts (cell types, stages of development, diseases etc) would be a valuable addition to public data sources, perhaps more than single cell expression data. For small molecule perturbations, we have consistently found that off-the-shelf small molecule embeddings perform very poorly on our small molecule benchmarks. Many of these models were trained to predict chemical properties, which may not connect closely to biological function. The field seems to lack a standard foundation model for small molecules that reflects biological function. Such a model may provide large benefits for predicting small molecule perturbations.

Another way to improve the results is fine-tuning. We have already shown that fine-tuning of individual models can improve results. Fine-tuning of integrated models may improve performance further, but this is a much more challenging and computationally demanding task. It remains to be seen if jointly fine-tuning multiple foundation models in a fusion framework can lead to further improvements. The risk of overfitting would be high without additional training data. To conclude, our comprehensive analysis indicates that FMs are indeed pushing the boundaries of our ability to accurately predict perturbations. Larger datasets and better FM methodologies may lead to very accurate models for this task.

4 Methods

4.1 Formulation of Log Fold-Change Regression

4.1.1 Notation

We start with a collection of N scRNA-seq profiles from a perturbation experiment in which K different perturbations P_1, \dots, P_K are tested. If we measure G genes for each cell, our observed expression matrix can then be written $X \in \mathbb{R}^{N \times G}$. Let X_i denote the i th row of X . We assume each row X_i has been total count normalized, scaled by 10^4 , and \log_{1p} transformed. Let S^{ctrl} be the set of indices for rows of X corresponding to control cells and let S^{P_k} be the set of indices for rows of X corresponding to cells receiving perturbation P_k . Thus, $|S^{P_1}| + \dots + |S^{P_K}| + |S^{\text{ctrl}}| = N$. Finally, we assume our N cells are partitioned into M batches based on some experimental condition. To describe this, let $S(b)$ be the set of indices of cells belonging to batch b . This means that $\sum_{b=1}^M |S(b)| = N$. Let

$$B_k = \{b : |S(b) \cap S^{\text{ctrl}}| > 0 \wedge |S(b) \cap S^{P_k}| > 0\} \quad (1)$$

denote the set of batches containing both control cells and cells that received perturbation P_k .

4.1.2 Definition of Ground Truth Labels

Our focus in this work is predicting the average treatment effect (ATE) of unseen perturbations. To account for batch effects, we weight the contribution of each batch equally. The result is a “batch aware” average treatment effect (BA-ATE). We use two variants, depending on whether the dataset’s batches are typically large (≥ 100) or small (< 100).

First, define the average control expression in batch b as

$$y_b = \frac{1}{|S^{\text{ctrl}} \cap S(b)|} \sum_{i \in S^{\text{ctrl}} \cap S(b)} X_i \quad (2)$$

where $y_b \in \mathbb{R}^G$. Similarly, define the average expression under perturbation P_k in batch b as

$$y'_{b,k} = \frac{1}{|S^{P_k} \cap S(b)|} \sum_{i \in S^{P_k} \cap S(b)} X_i \quad (3)$$

where $y'_{b,k} \in \mathbb{R}^G$.

Throughout the paper we refer to the BA-ATE as “log fold-change” (LFC). This is an abuse of terminology, but a useful one since BA-ATE is very similar in spirit. The differences are that BA-ATE (i) includes a batch adjustment, (ii) uses a \log_{1p} transform instead of a \log transform, and (iii) performs averaging after the \log_{1p} transform.

Datasets with large batches (Tahoe, Sciplex, Norman). Large batches allow us to accurately determine perturbation effects using perturbed and control cells from the same batch. We define the BA-ATE for large batches as

$$\Delta_k = \frac{1}{|B_k|} \sum_{b \in B_k} (y'_{b,k} - y_b). \quad (4)$$

Datasets with small batches (Essential). When batches are small, estimates of per-batch control expression are likely to be noisy. In this case, we first use all batches (whether or not there are matched perturbed cells) to estimate the global control expression for gene g :

$$y = \frac{1}{M} \sum_{b=1}^M y_b \quad (5)$$

where $y \in \mathbb{R}^G$. Then we compute the BA-ATE for small batches as

$$\Delta_k = \frac{1}{|B_k|} \sum_{b \in B_k} y'_{b,k} - y. \quad (6)$$

4.1.3 Metrics

Let $\Delta_k \in \mathbb{R}^G$ be the vector of per-gene treatment effects for perturbation P_k . Let $\hat{\Delta}_k \in \mathbb{R}^G$ be some model prediction for the same quantity. Our default performance metric is the simple L2 error, averaged over perturbations:

$$\frac{1}{K} \sum_k \|\Delta_k - \hat{\Delta}_k\|_2. \quad (7)$$

In Figure S11 we also consider other standard metrics, including MAE, MSE, Spearman correlation, and Pearson correlation.

4.2 Formulation of Differentially Expressed Gene (DEG) Classification

4.2.1 Notation

We begin with the same perturbation experiment setting and notation described in the LFC regression problem formulation in Section 4.1.1. We use parallel notation where possible (e.g. Δ_k for the effect of perturbation P_k) since the meaning is clear from context.

4.2.2 Definition of Ground Truth Labels

In this problem formulation, we aim to predict the differential expression status of each gene instead of the exact change in expression. We account for batch effects and weight the contributions of each batch equally.

For batch b , the differential expression results for perturbation P_k are computed as

$$y_{b,k} = \text{DE}(\{X_i : i \in S^{\text{ctrl}} \cap S(b)\}, \{X_i \in S^{P_k} \cap S(b)\}) \quad (8)$$

where the DE operator performs an independent differential expression test for each gene using the control cells $\{X_i : i \in S^{\text{ctrl}} \cap S(b)\}$ and the perturbed cells $\{X_i \in S^{P_k} \cap S(b)\}$. This results in a vector $y_{b,k} \in \{-1, 0, +1\}^G$, indicating whether each gene is downregulated (-1), upregulated ($+1$), or not significantly different (0).

Datasets with large batches (Tahoe, Norman, Sciplex). After computing per-batch differential expression vectors $y_{b,k}$, we produce a final ground truth perturbation effect $\Delta_k \in \{-1, 0, +1\}^G$ for perturbation P_k by performing majority vote over batches.

Datasets with small batches (Essential). For Essential, the batches are too small to obtain meaningful per-batch differential expression results. We therefore ignore batch structure when performing the differential expression testing, comparing all perturbed cells to all control cells:

$$\Delta_k = \text{DE}(\{X_i : i \in S^{\text{ctrl}}\}, \{X_i : i \in S^{P_k}\}). \quad (9)$$

Implementation details. All differential expression results were computed using Student's t-test with Benjamini-Hochberg multiple testing correction across genes. We used the implementation `rank_genes_groups` from Scanpy. We discard all (perturbation, batch) combinations with fewer than 30 cells. If no batches have more than 30 cells for a given perturbation, then that perturbation is discarded.

4.2.3 Metrics

We report macro F1 scores (Supplemental Methods).

4.3 Data

Here we describe the perturbation datasets used for embedding benchmarking.

For all datasets, we apply the same normalization scheme. We first normalize total counts to $1e4$. We then apply the \log_{1p} transformation to each count. This normalization is done as a preprocessing step, before all downstream calculations.

4.3.1 Norman

The Norman dataset is a genetic perturbation dataset (CRISPR activation, i.e. over-expression of the target gene) in one cell line, K562 [4]. The genes selected for perturbation were known to be involved in the regulation of cell growth. The dataset contains both single and dual perturbations. We use only the 105 single gene perturbations. This subset of the Norman dataset has 8 large batches (13k+ cells each), which are defined by GEM group.

4.3.2 Essential

The Essential dataset is a genetic perturbation dataset (CRISPR knockdown) of over 2000 “essential” genes per cell line for four cell lines (2387 for Hep-G2, Jurkat, and hTERT-RPE-1; 2054 for K-562.). There are 964451 single cells profiled. It has been compiled from two independent publications from the same lab [53, 54]. The Essential dataset has 56 small batches, which are defined by GEM group. There are 133784 unique (batch, perturbation) pairs, so on average there are 7.2 cells per perturbation and batch.

Note that the Essential dataset undergoes special preprocessing for the cross-context comparison with STATE (Supplemental Methods).

4.3.3 Sciplex

SciPlex3 [5] is a single cell chemical perturbation dataset. The raw expression matrix contains 799,317 single cells and 110,983 gene features, reflecting the full unfiltered set of detected transcripts. SciPlex3 profiles three human cancer cell lines: A549 lung adenocarcinoma, MCF7 breast adenocarcinoma, and K562 chronic myelogenous leukemia. Cells are treated with 188 distinct small molecule drugs as well as controls. Each drug is applied at four concentration levels: 10 nM, 100 nM, 1000 nM, and 10000 nM. In this work, we restrict the analysis to protein coding genes only. After filtering, the dataset contains 18,590 protein coding genes, which are used for all downstream modeling and evaluation. We further restrict all experiments to the highest dose level of 10000 nM for each drug. This dataset is organized into 52 medium sized batches, consisting of 154.4 cells on average for each (plate, drug, dose) combination (of which there are 4940).

4.3.4 Tahoe

The Tahoe-100M dataset is a chemical perturbation dataset generated by Tahoe Therapeutics [55]. It contains ~100M cells and measures ~60k genes. The cells are from 50 cell lines and profile 379 drugs, each at 3 dose levels. We used the highest dose - 5 uM - in our experiments. We excluded 5 cell lines (CVCL_1571, CVCL_1531, CVCL_1715, CVCL_1577, CVCL_1716) because their control cells had unusually low total counts. We also removed any compounds without SMILES string representations. The Tahoe dataset has 14 large batches, which are defined by the plate on which the cells were cultured. There are 15932 unique (plate, drug, dose) triplets, with an average of 6317.4 cell each.

Note that the Tahoe dataset undergoes special preprocessing for the cross-context comparison with STATE (Supplemental Methods).

4.4 Compute

Many of the methods used, studied, and tuned in this work required significant compute resources. Most notably, training and evaluating some of the advanced methods for single cell expression matching (Latent Diffusion, Schrödinger Bridge) required ~1000 GPU-hours each. Other parts of the study that relied on extensive compute

include the fine tuning of expression and interactome FMs and learning the integrated fusion model. Overall the study required approximately 5500 GPU-hours (~ 7.6 GPU-months). We used a combination of H100 and A100 GPUs for the work, depending on availability.

4.5 Embeddings for Genetic Perturbation Modeling

In the context of genetic perturbation modeling, we embed perturbations by embedding the gene that is being targeted. Below we detail the approaches we consider.

4.5.1 Expression

Unlike the other embedding categories, all of the expression-based gene embeddings are contextualized in the sense that they are based on control cell data from a given cell line.

Preliminaries: Fixed vs. variable gene set aggregation. The expression-based FMs described below are used to map a collection of single cells $X \in \mathbb{R}^{N \times G}$ to a collection of d -dimensional gene embeddings $Z \in \mathbb{R}^{d \times G}$. These single cells are typically the control cells for a particular cell line.

For some models (AIDO.Cell, scGPT, scPRINT) the gene set is the same for every cell. In this case, we obtain $Z' \in \mathbb{R}^{N \times d \times G}$ and simply compute

$$Z = \frac{1}{N} \sum_{i=1}^N Z'[i, :, :]. \quad (10)$$

However, there are other models (Geneformer, TranscriptFormer) for which the gene set used to encode a cell depends on the cell itself. Let $Q_g \subset \{1, \dots, N\}$ be the set of cell indices for which gene g is used to encode the cell (and thus an embedding for g exists). As we iterate through cells, we obtain tuples $(g, z_i(g))$ where $g \in \{1, \dots, G\}$ denotes a gene and $z_i(g)$ denotes the embedding of gene g from cell i . Then we compute

$$z(g) = \frac{1}{|Q_g|} \sum_{i \in Q_g} z_i(g) \quad (11)$$

and form $Z = \text{concat}([z(1), \dots, z(G)])$. Thus, some gene embeddings may be more stably estimated than others depending on the size of Q_g .

AIDO.Cell 3M/10M/100M [39]. AIDO.Cell is a family of transformer-based cellular foundation models, pretrained on 50 million cells spanning a wide range of tissues. AIDO.Cell performs dense self-attention over $\sim 18,000$ protein-coding genes, producing a d -dimensional contextualized embedding for each gene for each cell. To compute gene embeddings for a cell line, we compute the average gene embeddings over all control cells. The gene set is fixed. We also consider the learned positional encodings from the AIDO.Cell (3M) model as embeddings (called ‘‘Positional’’ in e.g. Figure 1) - this ablation allows us to measure the marginal benefit of the transformer layers.

Geneformer [28]. Trained on 30 million cells and adopts a language-model-inspired strategy. Instead of encoding all genes directly, it first ranks genes within each cell by expression and selects the top 2048. These “top-ranked genes” are treated as “tokens,” forming a cell-specific “sentence” that is processed by transformer layers. Given that this set of genes varies from cell-to-cell, we use the variable gene set aggregation method described above.

PCA. A simple and natural way to produce gene embeddings is to perform PCA on the cells in the control population. Formally, let $X \in \mathbb{R}^{N \times G}$ be a matrix of expression values for G genes over N control cells for some cell line. Choose an embedding dimension d . Then by performing PCA on X^T we can produce G embeddings of dimension d .

scGPT [30]. A transformer-based foundation model trained on 33 million cells that bins the expression values of genes before obtaining a gene-specific embedding per cell. The gene set is fixed.

scPRINT [56]. A cellular foundation model trained on 50 million cells designed to infer gene regulatory networks. It uses a bidirectional multi-head transformer and optimizes a zero-inflated negative binomial loss. The gene set is fixed.

TranscriptFormer [57]. A generative, multi-species foundation model trained on 112 million single cells from 12 species. It uses a fixed context length of 2048 “genes”, but this vocabulary varies depending on the species and context of the input. Given that this set of genes varies from cell-to-cell, we use the variable gene set aggregation method described above.

4.5.2 Prior Knowledge

DepMap [58]. An embedding may capture gene function without being useful for predicting the effect of a gene knockout. For instance, consider a gene whose loss can be compensated for by another. To construct embeddings that capture this sort of information, we collect gene essentiality measurements for over 400 cell lines from the DepMap Portal. This yields a matrix $A \in \mathbb{R}^{N \times G}$ where N is the number of cell lines and G is the number of genes for which essentiality scores are measured. To represent a gene, we simply use the corresponding column $z_k = A[:, \tau(k)]$ where $\tau(k)$ indexes the gene corresponding to perturbation P_k .

GenePT [59]. GenePT produces gene embeddings by taking text descriptions of genes from the biomedical literature and using an LLM (GPT-3.5) to extract embeddings of that text. We consider 6 variants of embeddings from GenePT, one based on NCBI gene cards and UniProt gene descriptions and 5 based on gene ontology information.

GenotypeVAE [60]. GenotypeVAE learns gene embeddings by training a variational autoencoder to learn latent embeddings of gene ontology annotation vectors. The original GenotypeVAE was trained with a latent dimension of 10. For the embeddings in this work, we retrained the model with a latent dimension of 128 to increase the capacity of the model. Additionally, we used the latest gene annotations from Functionome [61] to encode genes. All hyperparameters were identical to those from the original GenotypeVAE.

STRING GNN. We train a simple graph convolution network with residual skip connections using the link prediction objective on StringDB [62], a protein-protein interaction network. Node embeddings are randomly initialized. We used around 300 Optuna trials to tune over layers (2, 4, 6, 8), dimensionality (128, 256), dropout (0, 0.1, 0.2), learning rate (1e-3, 1e-4), weight decay (0, 0.01), batch size (1024, 2048, 4096, 8192, 16384), StringDB graph filter threshold (0.6, 0.7, 0.8, 0.9), and layer type (Graph Isomorphism Network (GIN), Graph Convolutional Network (GCN)). Optimal values are underlined. We performed a 90:10 train to validation split of the StringDB edges and selected the model with the lowest validation loss on the link prediction objective.

STRING Network [63]. Network-based embeddings were obtained by applying weighted node2vec to species-specific STRING protein-protein association networks, resulting in 128-dimensional node representations.

STRING Spectral. Starting from StringDB, we construct a weighted interaction network using only the experimental, coexpression, and database channels. Interactions were filtered to maintain a medium-confidence threshold, retaining only those with a combined weight greater than 400. Using the `SpectralEmbedding` function from scikit-learn, the resulting adjacency matrix was projected into a 128-dimensional space. This spectral approach utilizes Laplacian eigenmaps to transform the graph’s topological structure into Euclidean vectors, ensuring that proteins with functional associations are positioned closely together in the embedding space.

WaveGC [64]. WaveGC is a wavelet-based graph convolutional network. We use 2 layers, 3 wavelet scales, and 7 Chebyshev terms. The model was trained using the Adam optimizer with a learning rate of 1e-3 for 200 epochs. We select the model with the best validation loss. We use early stopping with a tolerance of 30 epochs. The model was trained on link prediction using the STRING network.

4.5.3 DNA

AIDO.DNA [33]. We use the AIDO.DNA model to compute gene embeddings based on the nucleotide sequence. For each gene, we run inference on a 4kbp window centered at the transcription start site. This yields per-nucleotide embeddings, which are then mean pooled to form gene embeddings.

4.5.4 Protein

In all cases below, multiple protein isoforms may be available for a given gene. In such cases, the embedding for a gene is the mean over embeddings of corresponding isoforms. For models that produce per-residue embeddings, protein embeddings are produced by mean pooling over residues.

AIDO.StructureTokenizer [65]. 300M parameter VQ-VAE model for protein structure tokenization.

AIDO.ProteinIF-16B [66]. 16B parameter protein language model based on a sparse mixture of experts.

AIDO.Protein-RAG-16B [35]. A multimodal protein language model that integrates Multiple Sequence Alignment (MSA) and structural data.

ESM2 [34]. 15B parameter protein language model trained on single protein sequences, without using MSA data.

STRING Sequence [67]. Protein sequence embeddings from the ProtT5, an encoder-decoder transformer model.

4.6 Embeddings for Chemical Perturbation Modeling

Molecular fingerprints. Fingerprints are fixed-length vector representations of molecules, obtained by hashing the presence or absence of specific chemical substructures (e.g., circular subgraphs, paths, atom environments). They are fast to compute and widely used in cheminformatics. We use several from RDKit [68]: MACCS, Topological, SECPF, ECPF:2, Avalon, and ErG.

SMILES-based foundation models. These models - MiniMol [69], ChemBERTa [36], Uni-Mol [70], MolT5 [71] - are large pretrained transformers that operate on either a SMILES string or a molecular structure graph. These models are trained with different supervised and self-supervised objectives, including contrastive learning between text descriptions of the drug and structure-based embeddings of the drug (MolT5), multi-target regression on chemistry-related labels from RDKit (ChemBERTa-MTR, MiniMol), masked atom prediction (Uni-Mol), and masked language modeling on the SMILES string (ChemBERTa-MLM),

Text embeddings. These embeddings (ChatGPT) use an off-the-shelf LLM (in our case, `text-embedding-3-small` from OpenAI) to embed textual metadata. In particular, we form a prompt that includes the name of the drug, the SMILES representation, any known targets, the mechanism of action (if known), and clinical trial status.

Target embeddings (FMs). Many small molecules have known or suspected gene targets. To put this into practice for a given chemical perturbation dataset, we first use an LLM to predict 3 genetic targets for each small molecule given the name of the drug. (We make no claim about the accuracy of these targets - we simply evaluate their utility on the downstream task of perturbation response prediction.) We use `claude-4-sonnet-20250514`. We then use expression FMs (scPRINT, AIDO.Cell 100M, TranscriptFormer) to embed those genes. For simplicity, we use the embeddings derived from the Norman dataset. Using the control cells from the chemical perturbation dataset itself may perform better. The embeddings of the three targets are concatenated.

Target embeddings (ablation). As an ablation on the FM-based target embeddings, we also experiment with encoding the targets as binary vectors (1 if a gene is a target, 0 otherwise; the number of genes is equal to the number of unique targets in the dataset). This method is denoted Targets Binary (Name). We also try a weighted version where we ask the LLM to return targets along with confidence weights: 1.0, 0.5, or 0.25. Then the target encoding replaces the "1" values with the corresponding weights. This method is named Targets Weighted (Name). Finally, we also try providing PubChem descriptions for each drug in addition to the drug name when querying for targets. This results in the methods Targets Binary (Name, PubChem) and Targets Weighted (Name, PubChem). We again use `claude-4-sonnet-20250514` for target prediction.

Affinity-based embeddings. Boltz-2 [72] is an open-source implementation of AlphaFold 3 that predicts 3D protein-ligand structures and estimates binding affinity.

In this work, we used Boltz to develop embeddings of small molecules based on their predicted interactions with various proteins.

For each small molecule, we co-folded and computed the predicted binding affinity with a set of proteins. This allowed us to create a vector where each entry corresponds to the predicted binding affinity between the molecule and a given protein. We can treat this vector as an embedding of the small molecule.

We consider two variants: fragment-based embeddings and protein-based embeddings. See Supplemental Methods for details.

4.7 Perturbation Prediction Models

4.7.1 Positive Controls

To establish best case performance, we considered several positive control methods.

Idealized baseline. Let $D \in \mathbb{R}^{K \times G}$ be the matrix of treatment effects for G genes over K perturbations. Crucially, let these K perturbations include both the train and test data. Choose an embedding dimension d . We perform PCA on D , resulting in K embeddings of dimension d .

Intuitively, these embeddings directly encode the similarity between all perturbations in the dataset, including the test set. Formally, the Eckhart-Young theorem [73] implies that these are the d -dimensional embeddings that compress D optimally (in terms of L2 error, and assuming a linear decoder). Note that nonlinear models could in theory produce d -dimensional embeddings that compress D with lower error, but even so we expect this baseline to be difficult to beat.

Experimental error. The idealized baseline described above ignores experimental noise. In this work we do not measure the true experimental error under repeated experiments ϵ_{true} . However, we do introduce a method to estimate a lower bound on ϵ_{true} using bootstrapping. (It is a lower bound because ϵ_{true} would include the error we estimate as well as other sources of error.) The basic idea is that the difference between replicates of the same measurement (which we call “experimental error” in a slight abuse of terminology) the minimum error we can reasonably expect to achieve with any algorithm trained on that data.

Let $X \in \mathbb{R}^{N \times G}$ be a matrix of expression measurements for G genes in N cells, some of which receive a perturbation and some of which are control cells. Given control and perturbed cells, we can compute the perturbation’s treatment effect which we will call $\Delta \in \mathbb{R}^G$.

By sampling with replacement from X and repeating M times, we can create a collection of M bootstrap samples $\{\Delta'_1, \dots, \Delta'_M\}$. These samples can be transformed to samples of the error in estimating y by computing

$$S_{\text{err}} = \{\|\Delta'_1 - \Delta\|_2, \dots, \|\Delta'_M - \Delta\|_2\}. \quad (12)$$

The samples in S_{err} approximate the distribution of the error in estimating y from repeated experiments. Then we define $\epsilon(q)$ - the experimental error at q - to be the q th quantile of S_{err} .

For example, if a model’s prediction $\hat{\Delta}$ satisfies $\|\hat{\Delta} - \Delta\|_2 \leq \epsilon(0.9)$, then the experimental error is higher than the model’s error 10% of the time. We believe this is a reasonable definition of predicting “within experimental error”. Throughout this work we use $q = 0.9$. In the same vein, we can recompute DEGs under resampling to get a distribution of F1 scores and define a corresponding “experimental error” limit for the DEG classification setting. In this case, since higher is better, we use $q = 0.1$.

Note that in practice we perform a hierarchical bootstrap, where we bootstrap over batches (M_{outer} trials) and cells within batches (M_{inner} trials) with the total number of trials being $M = M_{\text{outer}} \times M_{\text{inner}}$. See Supplemental Methods for details.

4.7.2 Negative Controls

To establish null performance, we considered several negative control methods which we expect to work poorly.

Train Mean. For LFC regression, the Train Mean is a model that predicts the mean response over all training perturbations. For DEG classification, the Train Mean samples predictions from the distribution of classes observed for each gene in the training set.

No Change. For any perturbation P_k , this method simply predicts that P_k has no effect. That is, we predict $\hat{\Delta}_k = 0$ for all k , meaning that the perturbed expression was identical to control expression (in the LFC regression setting) or that no genes are differentially expressed (in the DEG classification setting).

Random Embeddings. For each perturbation P_k (gene or small molecule), we define a random embedding z_k as follows. Given an embedding dimension d , we sample for each perturbation an embedding $z_k \sim \text{Uniform}([0, 1]^d)$. These embeddings carry no information about true similarity between perturbations and therefore cannot enable any generalization. Predictive models (e.g. kNN, Lasso) are fit using these embeddings under our standard benchmarking protocol.

4.7.3 Classical Machine Learning Methods

We tested several standard machine learning models including lasso, linear regression, and kNN regression. We detail our use of these models in Supplemental Methods, where we discuss embedding evaluation protocols.

4.7.4 Advanced Perturbation Prediction Methods

In this section we describe the advanced perturbation prediction methods we evaluate in this work. For consistency, all models are evaluated using our LFC regression formulation, even those that make single-cell predictions (Latent Diffusion, Flow Matching, Schrödinger Bridge). For each perturbation, control cells are sampled within each batch and passed through the model conditioned on the perturbation embedding to generate predicted perturbed single-cell profiles. These predictions are averaged to form per-batch predicted pseudobulks, which are then averaged across batches. The final prediction for each perturbation is obtained by subtracting the global control pseudobulk from the batch-averaged predicted pseudobulk, resulting in predictions $\hat{\Delta}_k$ for each perturbation P_k .

GEARS. GEARS [27] is a graph neural network where the edges are constructed roughly based on cell-ontology. Specifically, there is a cell-state encoder that is a GNN, and a perturbation encoder that is also a GNN. The training scheme is that one cell is sampled from the control distribution, and the two GNNs are used to predict the gene expression of the perturbed cell. This model is trained on one cell type at a time.

Latent Diffusion. Diffusion models are a powerful class of generative models that learn to iteratively transform a Gaussian random vector into a high quality data sample [44, 74]. (If we perform modeling in a latent space instead of data space, this is termed *latent* diffusion.) Applications to perturbation modeling include Squidiff [42] and dbDiffusion [75].

In general, diffusion models have a forward and reverse process. In the forward process, a data sample x_0 has small amounts of noise progressively added until the data sample is fully transformed into a Gaussian random noise sample x_T after T steps. In the reverse or generative process, the model $P_\theta(x_{t-1}|x_t)$ learns to predict x_{t-1} from x_t by training a neural network (parameterized by θ) to predict the noise that was added to x_0 to produce x_t . At prediction time, the generative model is iteratively applied to new Gaussian noise samples to generate data samples. Generation is conditioned on an embedding of the perturbation being applied. See Supplemental Methods for implementation details.

Flow Matching. Flow Matching models can be seen as a generalization of diffusion models where, instead of transforming a sample from a Gaussian distribution to the data distribution, the sample can be transformed from one arbitrary distribution to another [45]. Applications in perturbation modeling include CellFlow [41] and CellFlux [76].

In Flow Matching, the model is trained to learn the vector field between two distributions by sampling linear interpolants during training. Specifically, for each training sample, x_0 is sampled from the start distribution, x_1 is sampled from the target distribution, and $t \in [0, 1]$ is a sampled time point. A linear interpolant is drawn between the samples $x_t = (1-t)*x_0 + t*x_1$ and a derivative is calculated at the point $dx_t/dt = x_1 - x_0$. The model is trained with the mean squared error (MSE) objective to predict the velocity dx_t/dt from x_t and t . At prediction time, Flow Matching uses ODE solvers to transform the start distribution into the target distribution according to the vector field learned by the model. Generation is conditioned on an embedding of the perturbation being applied. See Supplemental Methods for implementation details.

Schrödinger Bridge. Schrödinger Bridge models are a further generalization of diffusion and Flow Matching models that draws from the strengths of both [46]. This method has been applied to perturbation modeling in DEPARTURES [77] and ARTEMIS [43]. Schrödinger Bridge models allow for arbitrary distribution to arbitrary distributions (like Flow Matching) but are stochastic in nature (like diffusion). We adopt the implementation of simulation-free Schrödinger Bridge models from Tong et al. [78], which builds on the aforementioned diffusion and Flow Matching models. Generation is conditioned on an embedding of the perturbation being applied. See Supplemental Methods for implementation details.

STATE. STATE [47] is a large transformer-based model trained using 100M perturbed cells. The model was developed for cross-context prediction, i.e. predicting

the effect of perturbations seen during training in cell lines not seen during training. STATE operates at the level of cell populations, consuming a set of control cells and a perturbation encoding and predicting a set of perturbed cells. Implementation details for our analysis of STATE can be found in Supplemental Methods.

4.8 Fine-Tuning

4.8.1 AIDO.Cell

Formulation (Indexing). Here we describe our approach to fine-tuning an AIDO.Cell model on the LFC regression task. Recall that we aim to predict $\Delta_k \in \mathbb{R}^G$, which is defined to be the vector of observed treatment effects for perturbation P_k . Let $X \in \mathbb{R}^{B \times G'}$ denote a batch of B control cell scRNA-seq profiles measured over G' genes. (Note that in general, it is not necessary to have $G = G'$.) Let $f_\theta : \mathbb{R}^{B \times G'} \rightarrow \mathbb{R}^{B \times G' \times d}$ denote the AIDO.Cell encoder (parameterized by θ), which consumes a batch of scRNA-seq profiles and produces a d -dimensional embedding for each gene and cell. Let's define a function m to compute the mean over the batch dimension and define $F_\theta(X) = m \circ f_\theta(X) \in \mathbb{R}^{G' \times d}$. Finally, we introduce a prediction head (e.g. a linear layer or an MLP) $g_\phi : \mathbb{R}^d \rightarrow \mathbb{R}^G$. We then define our perturbation response predictor as

$$\hat{\Delta}_k^{\phi, \theta} = g_\phi \circ (F_\theta(X)[\tau(k), :]) \in \mathbb{R}^G \quad (13)$$

where $\tau(k)$ indexes the gene corresponding to perturbation P_k . To fine-tune, we use stochastic gradient descent to optimize

$$\operatorname{argmin}_{\phi, \theta} \|\hat{\Delta}_k^{\phi, \theta} - \Delta_k\|_2. \quad (14)$$

Note that training was performed using data from only a single cell line, so the target output remains fixed regardless of the batch. A natural extension for future work would be to fine-tune across multiple cell lines so the model learns to differentiate perturbation effects in a cell type specific manner.

Formulation (In-Silico KO). Let $y \in \mathbb{R}^{G'}$ denote the average expression over all control cells. Suppose we want to make a prediction for perturbation P_k . Let $\tau(k)$ give the gene index corresponding to perturbation P_k . Let $Q \in \{0, 1\}^{G'}$ be a masking vector such that $Q[i] = 0$ if $i = \tau(k)$ and $Q[i] = 1$ otherwise. Let $f_\theta : \mathbb{R}^{G'} \rightarrow \mathbb{R}^{G' \times d}$ denote the AIDO.Cell encoder (parameterized by θ). Let $g_\phi : \mathbb{R}^d \rightarrow \mathbb{R}^G$ be a prediction head (parameterized by ϕ). Then we compute

$$\hat{\Delta}_k^{\phi, \theta} = [g_\phi \circ f_\theta(Qy)][\eta] \quad (15)$$

where $\eta \in \mathbb{R}^G$ is a set of indices that pick out the genes we want to predict. We then optimize

$$\operatorname{argmin}_{\phi, \theta} \|\hat{\Delta}_k^{\phi, \theta} - \Delta_k\|_2. \quad (16)$$

Implementation details. For indexing mode, we implement g_ϕ as an MLP with two hidden layers of sizes 1024 and 2048. For in-silico KO, we implement g_ϕ as an MLP with two hidden layers of sizes 64 and 32. We trained using LoRA for 20 epochs for indexing mode and 250 epochs for in-silico KO. Both models seem to have trained to convergence. We use a batch size of 8. In each batch, we randomly select 8 perturbations to update. The only hyperparameter optimization we performed was tuning the learning rate, which we tuned over $\{1e-2, 1e-3, 1e-4, 1e-5\}$ using the validation set. We explored both the Adam and Muon optimizers but did not observe a significant difference in performance. Within each of the 5 folds, we took 20% of the training data for validation.

4.8.2 STRING GNN

Formulation. We aim to predict observed treatment effects $\Delta_k \in \mathbb{R}^G$ for a perturbation P_k . Let W denote a graph over G' genes, with some node embeddings of dimension d' . Let $f_\theta : \mathbb{R}^{G' \times d'} \rightarrow \mathbb{R}^{G' \times d}$ denote our GNN, parameterized by θ . Let $g_\phi : \mathbb{R}^d \rightarrow \mathbb{R}^G$ be a prediction head, mapping a gene embedding to an estimate $\hat{\Delta}_k$ of Δ_k . Define $F_\theta(k|W) = f_\theta(W)[\tau(k), :]$ where $\tau(k)$ provides the index corresponding to perturbation P_k . Then our prediction for perturbation P_k is

$$\hat{\Delta}_k^{\phi, \theta} = g_\phi \circ F_\theta(k|W). \quad (17)$$

The training objective is

$$\operatorname{argmin}_{\phi, \theta} \|\hat{\Delta}_k^{\phi, \theta} - \Delta_k\|_2. \quad (18)$$

Implementation details. We initialize f_θ as the STRING GNN model we used to generate embeddings, which was selected using the validation performance on the link prediction task. We let g_ϕ be a linear projection. We follow the same 5-fold cross-validation splits used throughout the paper for Essential. For each fold, we split off 10% of the training perturbations for a validation set and train on the rest. We tune the learning rate (1e-3, 1e-4, 1e-5) and batch size (16, 32, 64) and select the model with the lowest validation loss. The optimal hyperparameters for each split can be found in Table S3.

4.9 Integrating FMs to Predict Perturbation Results

4.9.1 Preliminaries

Let $E_k \subset E$ denote the set of embedding sources that are valid for perturbation P_k , where $E = \{1, \dots, J\}$ is the set of all possible embedding sources. Let $Z_k = \{z_k^{(e)} : e \in E_k\}$ denote the set of valid embeddings for perturbation P_k , where $z_k^{(e)} \in \mathbb{R}^{d_e}$. Typically E_k is a strict subset of E . (For example, in the genetic perturbation setting we might use a scRNA-seq FM as one gene embedding source e . But if the gene corresponding to P_k was not used when training the FM, then $z_k^{(e)}$ is not defined.) Let L be the total number of cell lines under consideration. Note that some embeddings are

cell-line specific, so in general we should write $Z_{k,l}$ to acknowledge this dependence. We omit this unless necessary for clarity.

4.9.2 Simple Fusion Model

The aim of our embedding fusion model is to use the set of embeddings Z_k to predict the observed treatment effect Δ_k (for either the LFC regression or DEG classification case). We propose an attention-based fusion head for integrating embeddings, which we describe below. This model is trained jointly over multiple cell lines.

Architecture. In the first stage of our model, each embedding $z_k^{(e)}$ is passed through a preprocessing stage:

$$Q_k = \{\text{Dropout}(f_{\theta(e)}(z_k^{(e)}), p = 0.1) : e \in E_k\} \quad (19)$$

where $f_{\theta(e)} : \mathbb{R}^{d_e} \rightarrow \mathbb{R}^d$ is a per-source projection model which maps embeddings of type e from dimension d_e to a common dimension d . We parameterize the cell line using a learnable, randomly initialized vector $z_{\text{line}} \in \mathbb{R}^d$, which is preprocessed as

$$q_{\text{line}} = \text{Dropout}(\text{LayerNorm}(z_{\text{line}}), p = 0.1). \quad (20)$$

We also create a CLS token q_{CLS} . We form our input set for perturbation P_k as

$$S_{k,l} = Q_k \cup \{q_{\text{CLS}}\} + q_{\text{line}(l)} \quad (21)$$

where addition is performed elementwise. Note that we do not explicitly inject source identity information into these tokens because the bias terms in the models $f_{\theta(e)}$ are able to perform that function.

Let $T_\theta : \mathbb{R}^{L \times d} \rightarrow \mathbb{R}^{L \times d}$ denote a transformer model and let $g_\phi : \mathbb{R}^d \rightarrow \mathbb{R}^G$ denote a prediction head. We integrate information across embeddings by computing $R = T_\theta(S_{k,l})$. Finally, we make predictions as $\hat{\Delta}_k = g_\phi(r_{\text{CLS}})$ where r_{CLS} is the contextualized output (and thus a row of R) corresponding to q_{CLS} .

Implementation details. We fix the transformer’s $d_{\text{model}} = d = 100$ and use 5 attention heads. We train with an L2 loss using a batch size of 64 and the Adam optimizer. We use Optuna to tune over the depth of T_θ (ranging from 1 to 8 layers) and the learning rate (ranging from 1e-5 to 1e-3, searching logarithmically). We tune on one fold and use the same hyperparameters for all other folds.

We also tune over a binary variable that toggles between the “global” cell line injection style presented above and an alternative approach where we append the cell line token to the input set. For the LFC regression setting on Essential (where fusion was quite successful), we found that the former consistently worked better than the latter.

We used 100 Optuna trials for all cases except Tahoe DEG, where we stopped tuning after 50 trials. In Tahoe DEG, we observed no improvement past the 22nd trial.

4.9.3 Full Fusion Model

This model is applied only to the Essential LFC task since other datasets did not show improvements with the simple fusion model. It is generally more complex and harder to tune, but it demonstrates that architectural innovations are available that can push the performance of multimodal fusion higher. This model is trained jointly over multiple cell lines. See Supplemental Methods for complete details.

5 Declarations

Competing Interests

All authors are current or former employees of GenBio AI and may hold a financial interest in the company. GenBio AI provided the funding and resources for this research. EL performed this work as part of her consulting role for GenBio AI, not at Stanford University.

Data and Code Availability

Data and code for this work can be found on GitHub:

<https://github.com/genbio-ai/foundation-models-perturbation>

References

- [1] Molinelli, E.J., Korkut, A., Wang, W., Miller, M.L., Gauthier, N.P., Jing, X., Kaushik, P., He, Q., Mills, G., Solit, D.B., *et al.*: Perturbation biology: inferring signaling networks in cellular systems. *PLoS computational biology* **9**(12), 1003290 (2013)
- [2] Ji, Y., Lotfollahi, M., Wolf, F.A., Theis, F.J.: Machine learning for perturbational single-cell omics. *Cell Systems* **12**(6), 522–537 (2021)
- [3] Gavriilidis, G.I., Vasileiou, V., Orfanou, A., Ishaque, N., Psomopoulos, F.: A mini-review on perturbation modelling across single-cell omic modalities. *Computational and Structural Biotechnology Journal* **23**, 1886 (2024)
- [4] Norman, T.M., Horlbeck, M.A., Replogle, J.M., Ge, A.Y., Xu, A., Jost, M., Gilbert, L.A., Weissman, J.S.: Exploring genetic interaction manifolds constructed from rich single-cell phenotypes. *Science* **365**(6455), 786–793 (2019)
- [5] Srivatsan, S.R., McFaline-Figueroa, J.L., Ramani, V., Saunders, L., Cao, J., Packer, J., Pliner, H.A., Jackson, D.L., Daza, R.M., Christiansen, L., *et al.*: Massively multiplex chemical transcriptomics at single-cell resolution. *Science* **367**(6473), 45–51 (2020)
- [6] Barretina, J., Caponigro, G., Stransky, N., Venkatesan, K., Margolin, A.A., Kim, S., Wilson, C.J., Lehár, J., Kryukov, G.V., Sonkin, D., *et al.*: The cancer cell line

encyclopedia enables predictive modelling of anticancer drug sensitivity. *Nature* **483**(7391), 603–607 (2012)

- [7] Yang, W., Soares, J., Greninger, P., Edelman, E.J., Lightfoot, H., Forbes, S., Bindal, N., Beare, D., Smith, J.A., Thompson, I.R., *et al.*: Genomics of drug sensitivity in cancer (gdsc): a resource for therapeutic biomarker discovery in cancer cells. *Nucleic acids research* **41**(D1), 955–961 (2012)
- [8] Hafner, M., Niepel, M., Chung, M., Sorger, P.K.: Growth rate inhibition metrics correct for confounders in measuring sensitivity to cancer drugs. *Nature methods* **13**(6), 521–527 (2016)
- [9] Bray, M.-A., Singh, S., Han, H., Davis, C.T., Borgeson, B., Hartland, C., Kost-Alimova, M., Gustafsdottir, S.M., Gibson, C.C., Carpenter, A.E.: Cell painting, a high-content image-based assay for morphological profiling using multiplexed fluorescent dyes. *Nature protocols* **11**(9), 1757–1774 (2016)
- [10] Koleti, A., Terryn, R., Stathias, V., Chung, C., Cooper, D.J., Turner, J.P., Vidović, D., Forlin, M., Kelley, T.T., D’Urso, A., *et al.*: Data portal for the library of integrated network-based cellular signatures (lincs) program: integrated access to diverse large-scale cellular perturbation response data. *Nucleic acids research* **46**(D1), 558–566 (2018)
- [11] Lamb, J., Crawford, E.D., Peck, D., Modell, J.W., Blat, I.C., Wrobel, M.J., Lerner, J., Brunet, J.-P., Subramanian, A., Ross, K.N., *et al.*: The connectivity map: using gene-expression signatures to connect small molecules, genes, and disease. *science* **313**(5795), 1929–1935 (2006)
- [12] Dixit, A., Parnas, O., Li, B., Chen, J., Fulco, C.P., Jerby-Arnon, L., Marjanovic, N.D., Dionne, D., Burks, T., Raychowdhury, R., *et al.*: Perturb-seq: dissecting molecular circuits with scalable single-cell rna profiling of pooled genetic screens. *cell* **167**(7), 1853–1866 (2016)
- [13] Ideker, T., Galitski, T., Hood, L.: A new approach to decoding life: systems biology. *Annual review of genomics and human genetics* **2**(1), 343–372 (2001)
- [14] Menden, M.P., Iorio, F., Garnett, M., McDermott, U., Benes, C.H., Ballester, P.J., Saez-Rodriguez, J.: Machine learning prediction of cancer cell sensitivity to drugs based on genomic and chemical properties. *PLoS one* **8**(4), 61318 (2013)
- [15] Hwang, T.J., Carpenter, D., Lauffenburger, J.C., Wang, B., Franklin, J.M., Kesselheim, A.S.: Failure of investigational drugs in late-stage clinical development and publication of trial results. *JAMA internal medicine* **176**(12), 1826–1833 (2016)
- [16] Weinstein, J.N., Kohn, K.W., Grever, M.R., Viswanadhan, V.N., Rubinstein, L.V., Monks, A.P., Scudiero, D.A., Welch, L., Koutsoukos, A.D., Chiousa, A.J.,

- et al.*: Neural computing in cancer drug development: predicting mechanism of action. *Science* **258**(5081), 447–451 (1992)
- [17] Staunton, J.E., Slonim, D.K., Collier, H.A., Tamayo, P., Angelo, M.J., Park, J., Scherf, U., Lee, J.K., Reinhold, W.O., Weinstein, J.N., *et al.*: Chemosensitivity prediction by transcriptional profiling. *Proceedings of the National Academy of Sciences* **98**(19), 10787–10792 (2001)
- [18] Gardner, T.S., Di Bernardo, D., Lorenz, D., Collins, J.J.: Inferring genetic networks and identifying compound mode of action via expression profiling. *Science* **301**(5629), 102–105 (2003)
- [19] Tegner, J., Yeung, M.S., Hastie, T., Collins, J.J.: Reverse engineering gene networks: integrating genetic perturbations with dynamical modeling. *Proceedings of the National Academy of Sciences* **100**(10), 5944–5949 (2003)
- [20] Zou, H., Hastie, T.: Regularization and variable selection via the elastic net. *Journal of the Royal Statistical Society Series B: Statistical Methodology* **67**(2), 301–320 (2005)
- [21] Iorio, F., Knijnenburg, T.A., Vis, D.J., Bignell, G.R., Menden, M.P., Schubert, M., Aben, N., Gonçalves, E., Barthorpe, S., Lightfoot, H., *et al.*: A landscape of pharmacogenomic interactions in cancer. *Cell* **166**(3), 740–754 (2016)
- [22] Wang, L., Li, X., Zhang, L., Gao, Q.: Improved anticancer drug response prediction in cell lines using matrix factorization with similarity regularization. *BMC cancer* **17**(1), 513 (2017)
- [23] Suphavitai, C., Bertrand, D., Nagarajan, N.: Predicting cancer drug response using a recommender system. *Bioinformatics* **34**(22), 3907–3914 (2018)
- [24] Rampásek, L., Hidru, D., Smirnov, P., Haibe-Kains, B., Goldenberg, A.: Dr. vae: improving drug response prediction via modeling of drug perturbation effects. *Bioinformatics* **35**(19), 3743–3751 (2019)
- [25] Lotfollahi, M., Wolf, F.A., Theis, F.J.: scgen predicts single-cell perturbation responses. *Nature methods* **16**(8), 715–721 (2019)
- [26] Lotfollahi, M., Klimovskaia Susmelj, A., De Donno, C., Hetzel, L., Ji, Y., Ibarra, I.L., Srivatsan, S.R., Naghipourfar, M., Daza, R.M., Martin, B., *et al.*: Predicting cellular responses to complex perturbations in high-throughput screens. *Molecular systems biology* **19**(6), 11517 (2023)
- [27] Roohani, Y., Huang, K., Leskovec, J.: Predicting transcriptional outcomes of novel multigene perturbations with gears. *Nature Biotechnology* **42**(6), 927–935 (2024)
- [28] Theodoris, C.V., Xiao, L., Chopra, A., Chaffin, M.D., Al Sayed, Z.R., Hill, M.C.,

- Mantineo, H., Brydon, E.M., Zeng, Z., Liu, X.S., *et al.*: Transfer learning enables predictions in network biology. *Nature* **618**(7965), 616–624 (2023)
- [29] Hao, M., Gong, J., Zeng, X., Liu, C., Guo, Y., Cheng, X., Wang, T., Ma, J., Zhang, X., Song, L.: Large-scale foundation model on single-cell transcriptomics. *Nature methods* **21**(8), 1481–1491 (2024)
- [30] Cui, H., Wang, C., Maan, H., Pang, K., Luo, F., Duan, N., Wang, B.: scgpt: toward building a foundation model for single-cell multi-omics using generative ai. *Nature methods* **21**(8), 1470–1480 (2024)
- [31] Wu, Y., Wershof, E., Schmon, S.M., Nassar, M., Osiński, B., Eksi, R., Yan, Z., Stark, R., Zhang, K., Graepel, T.: Perturbench: Benchmarking machine learning models for cellular perturbation analysis. *arXiv preprint arXiv:2408.10609* (2024)
- [32] Ahlmann-Eltze, C., Huber, W., Anders, S.: Deep-learning-based gene perturbation effect prediction does not yet outperform simple linear baselines. *Nature Methods* **22**(8), 1657–1661 (2025)
- [33] Ellington, C.N., Sun, N., Ho, N., Tao, T., Mahbub, S., Li, D., Zhuang, Y., Wang, H., Song, L., Xing, E.P.: Accurate and general dna representations emerge from genome foundation models at scale. *bioRxiv*, 2024–12 (2024)
- [34] Lin, Z., Akin, H., Rao, R., Hie, B., Zhu, Z., Lu, W., Smetanin, N., Verkuil, R., Kabeli, O., Shmueli, Y., *et al.*: Evolutionary-scale prediction of atomic-level protein structure with a language model. *Science* **379**(6637), 1123–1130 (2023)
- [35] Li, P., Cheng, X., Song, L., Xing, E.: Retrieval augmented protein language models for protein structure prediction. *bioRxiv*, 2024–12 (2024)
- [36] Chithrananda, S., Grand, G., Ramsundar, B.: Chemberta: large-scale self-supervised pretraining for molecular property prediction. *arXiv preprint arXiv:2010.09885* (2020)
- [37] Wenteler, A., Occhetta, M., Branson, N., Huebner, M., Curean, V., Dee, W., Connell, W., Hawkins-Hooker, A., Chung, S.P., Ektefaie, Y., *et al.*: Perteval-scfm: benchmarking single-cell foundation models for perturbation effect prediction. *bioRxiv*, 2024–10 (2024)
- [38] Littman, R., Levine, J., Maleki, S., Lee, Y., Ermakov, V., Qiu, L., Wu, A., Huang, K., Lopez, R., Scalia, G., Biancalani, T., Richmond, D., Regev, A., Hütter, J.-C.: Gene-embedding-based prediction and functional evaluation of perturbation expression responses with presage. *bioRxiv* (2025)
- [39] Ho, N., Ellington, C.N., Hou, J., Addagudi, S., Mo, S., Tao, T., Li, D., Zhuang, Y., Wang, H., Cheng, X., Song, L., Xing, E.P.: Scaling dense representations for single cell with transcriptome-scale context. *bioRxiv* (2024)

- [40] Wang, W., Yang, F., Fang, Y., Tang, D., Huang, J., Lu, H., Yao, J.: scbert as a large-scale pretrained deep language model for cell type annotation of single-cell rna-seq data. *Nature Machine Intelligence* **4**, 852–866 (2022)
- [41] Klein, D., Fleck, J.S., Bobrovskiy, D.M., Zimmermann, L., Becker, S., Palma, A., Dony, L., Tejada-Lapuerta, A., Huguet, G., Lin, H.-C., Azbukina, N., Sanchís-Calleja, F., Uscidda, T., Szalata, A., Gander, M., Regev, A., Treutlein, B., Camp, J.G., Theis, F.J.: Cellflow enables generative single-cell phenotype modeling with flow matching. *bioRxiv* (2025)
- [42] He, S., Zhu, Y., Tavakol, D.N., Ye, H., Lao, Y.-H., Zhu, Z., Xu, C., Chauhan, S., Garty, G., Tomer, R., Vunjak-Novakovic, G., Zou, J.Y., Azizi, E., Leong, K.W.: Squidiff: predicting cellular development and responses to perturbations using a diffusion model. *Nature Methods* **23**, 65–77 (2025)
- [43] Alatkari, S.A., Wang, D.: Artemis integrates autoencoders and schrödinger bridges to predict continuous dynamics of gene expression, cell population, and perturbation from time-series single-cell data. *Bioinformatics* **41**, 189–197 (2025)
- [44] Sohl-Dickstein, J.N., Weiss, E.A., Maheswaranathan, N., Ganguli, S.: Deep unsupervised learning using nonequilibrium thermodynamics. *ArXiv abs/1503.03585* (2015)
- [45] Lipman, Y., Chen, R.T.Q., Ben-Hamu, H., Nickel, M., Le, M.: Flow matching for generative modeling. *ArXiv abs/2210.02747* (2022)
- [46] Bortoli, V.D., Thornton, J., Heng, J., Doucet, A.: Diffusion schrödinger bridge with applications to score-based generative modeling. *ArXiv abs/2106.01357* (2021)
- [47] Adduri, A., Gautam, D., Bevilacqua, B., Imran, A., Shah, R., Naghipourfar, M., Teyssier, N.B., Ilango, R., Nagaraj, S., Dong, M., Ricci-Tam, C., Carpenter, C., Subramanyam, V., Winters, A., Tirukkovular, S., Sullivan, J., Plosky, B.S., Eraslan, B., Youngblut, N.D., Leskovec, J., Gilbert, L.A., Konermann, S., Hsu, P.D., Dobin, A., Burke, D.P., Goodarzi, H., Roohani, Y.H.: Predicting cellular responses to perturbation across diverse contexts with state. *bioRxiv* (2025)
- [48] Morgan, H.L.: The generation of a unique machine description for chemical structures—a technique developed at chemical abstracts service. *Journal of Chemical Documentation* **5**, 107–113 (1965)
- [49] Rogers, D., Hahn, M.: Extended-connectivity fingerprints. *Journal of chemical information and modeling* **50** **5**, 742–54 (2010)
- [50] Bohacek, R.S., McMartin, C., Guida, W.C.: The art and practice of structure-based drug design: a molecular modeling perspective. *Medicinal research reviews* **16**(1), 3–50 (1996)

- [51] Märtens, K., Donovan-Maiye, R., Ferkinghoff-Borg, J.: Enhancing generative perturbation models with llm-informed gene embeddings. In: ICLR 2024 Workshop on Machine Learning for Genomics Explorations (2024)
- [52] Chen, Y., Zou, J.: Genept: Leveraging genept embeddings for gene perturbation prediction. *bioRxiv*, 2024–10 (2024)
- [53] Replogle, J.M., Saunders, R.A., Pogson, A.N., Hussmann, J.A., Lenail, A., Guna, A., Mascibroda, L., Wagner, E.J., Adelman, K., Lithwick-Yanai, G., Iremadze, N., Oberstrass, F., Lipson, D., Bonnar, J.L., Jost, M., Norman, T.M., Weissman, J.S.: Mapping information-rich genotype-phenotype landscapes with genome-scale perturb-seq. *Cell* **185**(14), 2559–257528 (2022) <https://doi.org/10.1016/j.cell.2022.05.013>
- [54] Nadig, A., Replogle, J.M., Pogson, A.N., Murthy, M., Mccarroll, S., Weissman, J.S., Robinson, E.B., O’Connor, L.J.: Transcriptome-wide analysis of differential expression in perturbation atlases. *Nature Genetics* **57**, 1228–1237 (2025)
- [55] Zhang, J., Ubas, A.A., Borja, R., Svensson, V., Thomas, N., Thakar, N., Lai, I., Winters, A., Khan, U., Jones, M.G., Thompson, J.D., Tran, V.K., Pangallo, J., Papalexi, E., Sapre, A.A., Nguyen, H., Sanderson, O., Nigos, M., Kaplan, O., Schroeder, S., Hariadi, B., Marrujo, S., Salvino, C.C.A., Olivares, G.G., Koehler, R., Geiss, G., Rosenberg, A.B., Roco, C.M., Merico, D., Alidoust, N., Goodarzi, H., Yu, J.: Tahoe-100m: A giga-scale single-cell perturbation atlas for context-dependent gene function and cellular modeling. *bioRxiv* (2025)
- [56] Kalfon, J., Samaran, J., Peyré, G., Cantini, L.: scprint: pre-training on 50 million cells allows robust gene network predictions. *Nature Communications* **16** (2024)
- [57] Pearce, J.D., Simmonds, S.E., Mahmoudabadi, G., Krishnan, L., Palla, G., Istrate, A.-M., Tarashansky, A., Nelson, B., Valenzuela, O., Li, D., Quake, S.R., Karaletsos, T.: A cross-species generative cell atlas across 1.5 billion years of evolution: The transcriptformer single-cell model. *bioRxiv* (2025)
- [58] Broad Institute: DepMap Portal. <https://depmap.org/portal/>
- [59] Chen, Y.T., Zou, J.: Genept: A simple but effective foundation model for genes and cells built from chatgpt. *bioRxiv* (2023)
- [60] Yu, H., Qian, W., Song, Y., Welch, J.D.: Perturbnet predicts single-cell responses to unseen chemical and genetic perturbations. *Molecular Systems Biology* **21**, 960–982 (2025)
- [61] Feuermann, M., Mi, H., Gaudet, P., Muruganujan, A., Lewis, S., Ebert, D., Mushayahama, T., Hi, S.A.J.S.J.M.H.J.N.L.D.P.A.B.C.C.D.H., Alexander, S.A., Balhoff, J.P., Carbon, S., Cherry, J.M., Drabkin, H.J., Harris, N.L., Hill, D.P., Lee, R., Logie, C., Moxon, S.A.T., Mungall,

C.J., Sternberg, P.W., Auken, K.M.V., Siegele, J.D.A.R., Ramsey, J., Siegele, D.A., Fey, R.L.P.C., Chisholm, R.L., Fey, P., Nadendla, M.S.G., Giglio, M., Nadendla, S., Marygold, G.H.N.H.P.V.S.A.A.B.G., Antonazzo, G., Attrill, H., Brown, N.H., Garapati, P.V., Marygold, S.J., Kadhum, S.H.P.D.L.M.N.M.C.S.A.A.B.E.G.H., Ahmed, S.H., Asanitthong, P., Buitrago, D.L., Erdol, M.N., Gage, M.C., Huang, S., Kadhum, M.A., Li, K.Y.C., Long, M., Michalak, A., Pesala, A., Pritazahra, A., Saverimuttu, S.C.C., Su, R., Xu, Q., Lovering, R.C., Smith, J.K.L.M.E.L.D.C.B.C.C.D.N.S., Blake, J., Christie, K.R., Corbani, L.E., Dolan, M.E., Ni, L., Sitnikov, D., Smith, C.L., Wood, M.K.V.L.-R.R., Lera-Ramírez, M., Rutherford, K., Wood, V., D'Eustachio, P., Laul, W.M.J.L.M.R.G.T.M.L.A.E.D.D.P.D.H.K.K., Demos, W.M., Pons, J.L.D., Dwinell, M.R., Hayman, G.T., Kaldunski, M.L., Kwitek, A.E., Laulederkind, S.J.F., Smith, J.R., Tutaj, M.A., Vedi, M., Wang, S.-J., Wong, S.R.K.S.R.R.S.M.S.S.E.E.K.M.N.S.W., Engel, S.R., Karra, K., Miyasato, S.R., Nash, R.S., Skrzypek, M.S., Weng, S., Wong, E.D., Ch, T.M.C.À.T.N.J.J.E.M.A.A.-A.B.B.B.B., Achsel, T., Andres-Alonso, M., Bagni, C., Bayés, À., Biederer, T., Brose, N., Chua, J.J., Coba, M.P., Cornelisse, L.N., Juan-Sanz, J., Goldschmidt, H.L., Gundelfinger, E.D., Hukanir, R.L., Imig, C., Jahn, R., Jung, H., Kaeser, P.S., Kim, E., Koopmans, F., Kreutz, M.R., Lipstein, N., MacGillavry, H.D., McPherson, P.S., O'Connor, V., Pielot, R., Ryan, T.A., Sala, C., Sheng, M., Smalla, K.-H., Smit, A.B., Toonen, R.F.G., Weering, J.R.T., Verhage, M., Verpelli, C., Reiser, E.T.Z.L.B.B., Bakker, E., Berardini, T.Z., Reiser, L., Breu, A.K.G.M.-C.E.L.A.A.A.-P.B.B., Auchincloss, A.H., Axelsen, K.B., Argoud-Puy, G., Blatter, M.-C., Boutet, E., Breuza, L., Bridge, A.J., Casals-Casas, C., Coudert, E., Estreicher, A., Famiglietti, M.L., Gos, A., Gruaz-Gumowski, N., Hulo, C., Hyka-Nouspikel, N., Jungo, F., Mercier, P., Lieberherr, D., Masson, P., Morgat, A., Pedruzzi, I., Pourcel, L., Poux, S., Rivoire, C., Sundaram, S., Ishtiaq, E.H.P.A.R.A.Y.M.M.B.-B.B.-A.-J.D.I., Bowler-Barnett, E.H., Bye-A-Jee, H., Denny, P., Ignatchenko, A., Ishtiaq, R., Lock, A., Lussi, Y., Magrane, M., Martin, M.J., Orchard, S., Raposo, P., Speretta, E., Tyagi, N., Warner, K., Zaru, R., Raciti, J.S.D.C.D., Chan, J., Diamantakis, S., Raciti, D., Zorn, M.C.V.A.F.J.-Z.P., Fisher, M., James-Zorn, C., Ponferrada, V.G., Zorn, A., Westerfield, S.L.M.R.R., Ramachandran, S., Ruzicka, L., Westerfield, M., Thomas, P.D.: A compendium of human gene functions derived from evolutionary modelling. *Nature* **640**, 146–154 (2025)

- [62] Szklarczyk, D., Kirsch, R., Koutrouli, M., Nastou, K.C., Mehryary, F., Hachilif, R., Gable, A.L., Fang, T., Doncheva, N.T., Pyysalo, S., Bork, P., Jensen, L.J., Mering, C.: The string database in 2023: protein–protein association networks and functional enrichment analyses for any sequenced genome of interest. *Nucleic Acids Research* **51**, 638–646 (2022)
- [63] Hu, D., Szklarczyk, D., Mering, C., Jensen, L.J.: Space: String proteins as complementary embeddings. *Bioinformatics* **41** (2024)

- [64] Liu, N., He, X., Laurent, T., Giovanni, F.D., Bronstein, M.M., Bresson, X.: A general graph spectral wavelet convolution via chebyshev order decomposition. In: International Conference on Machine Learning (2024). <https://api.semanticscholar.org/CorpusID:269982678>
- [65] Zhang, J., Meynard-Piganeau, B., Gong, J., Cheng, X., Luo, Y., Ly, H., Song, L., Xing, E.P.: Balancing locality and reconstruction in protein structure tokenizer. bioRxiv (2024)
- [66] Sun, N., Zou, S., Tao, T., Mahbub, S., Li, D., Zhuang, Y., Wang, H., Cheng, X., Song, L., Xing, E.P.: Mixture of experts enable efficient and effective protein understanding and design. bioRxiv (2024)
- [67] Elnaggar, A., Heinzinger, M., Dallago, C., Rehawi, G., Wang, Y., Jones, L., Gibbs, T., Fehér, T.B., Angerer, C., Steinegger, M., Bhowmik, D., Rost, B.: Prottrans: Towards cracking the language of life's code through self-supervised deep learning and high performance computing. bioRxiv (2020)
- [68] Landrum, G.: RDKit: Open-source cheminformatics. <https://www.rdkit.org>
- [69] Kläser, K., Banaszewski, B.z., Maddrell-Mander, S., McLean, C., Müller, L., Parviz, A., Huang, S., Fitzgibbon, A.W.: Minimol: A parameter-efficient foundation model for molecular learning. ArXiv **abs/2404.14986** (2024)
- [70] Zhou, G., Gao, Z., Ding, Q., Zheng, H., Xu, H., Wei, Z., Zhang, L., Ke, G.: Uni-mol: A universal 3d molecular representation learning framework. In: International Conference on Learning Representations (2023). <https://api.semanticscholar.org/CorpusID:259298651>
- [71] Edwards, C.N., Lai, T., Ros, K., Honke, G., Ji, H.: Translation between molecules and natural language. ArXiv **abs/2204.11817** (2022)
- [72] Passaro, S., Corso, G., Wohllwend, J., Reveiz, M., Thaler, S., Somnath, V.R., Getz, N., Portnoi, T., Roy, J., Stärk, H., Kwabi-Addo, D., Beaini, D., Jaakkola, T., Barzilay, R.: Boltz-2: Towards accurate and efficient binding affinity prediction. bioRxiv (2025)
- [73] Eckart, C., Young, G.M.: The approximation of one matrix by another of lower rank. *Psychometrika* **1**, 211–218 (1936)
- [74] Ho, J., Jain, A., Abbeel, P.: Denoising diffusion probabilistic models. ArXiv **abs/2006.11239** (2020)
- [75] Shang, E., Wei, Y., Roeder, K.: Predicting the unseen: a diffusion-based debiasing framework for transcriptional response prediction at single-cell resolution. bioRxiv (2025)

- [76] Zhang, Y., Su, Y., Wang, C., Li, T., Wefers, Z., Nirschl, J.J., Burgess, J., Ding, D., Lozano, A., Lundberg, E., Yeung-Levy, S.: Cellflow: Simulating cellular morphology changes via flow matching. ArXiv [abs/2502.09775](#) (2025)
- [77] Chi, C., Huang, Y., Xia, J., Zheng, J., Liu, Y., Zang, Z., Li, S.Z.: Departures: Distributional transport for single-cell perturbation prediction with neural schrödinger bridges. ArXiv [abs/2511.13124](#) (2025)
- [78] Tong, A., Malkin, N., Fatras, K., Atanackovic, L., Zhang, Y., Huguet, G., Wolf, G., Bengio, Y.: Simulation-free schrödinger bridges via score and flow matching. In: International Conference on Artificial Intelligence and Statistics (2023). <https://api.semanticscholar.org/CorpusID:259375451>