Deep learning-based predictions of gene perturbation effects do not yet outperform simple linear methods

Constantin Ahlmann-Eltze[⋆,†], Wolfgang Huber[⋆], and Simon Anders[†]

[⋆]*Genome Biology Unit, EMBL, Heidelberg, Germany.*
[†]*Bioquant, University of Heidelberg, Germany*

16th of September 2024

Abstract

Advanced deep-learning methods, such as transformer-based foundation models, promise to learn representations of biology that can be employed to predict *in silico* the outcome of unseen experiments, such as the effect of genetic perturbations on the transcriptomes of human cells. To see whether current models already reach this goal, we benchmarked two state-of-the-art foundation models and one popular graph-based deep learning framework against deliberately simplistic linear models in two important use cases: For combinatorial perturbations of two genes for which only data for the individual single perturbations have been seen, we find that a simple additive model outperformed the deep learning-based approaches. Also, for perturbations of genes that have not yet been seen, but which may be "interpolated" from biological similarity or network context, a simple linear model performed as good as the deep learning-based approaches. Our results indicate that the biology learned by current models does not yet allow to extrapolate beyond the specific experimental conditions of the training data, and that recent claims to the contrary may have been premature.

Contact: constantin.ahlmann@embl.de

Introduction

The success of large language models in knowledge representation has spurned research efforts to apply foundation models to biology.[1,2,3] The expectation is that such models can ingest enormous amounts of biological data and learn enough about biological systems, with their gene regulatory networks, metabolic pathways, protein complexes, and cellular architecture, to then reliably *generate* the outcome of unseen experiments—in analogy to generative models that generate meaningful texts or images upon a user's prompt. Such capabilities could replace expensive experiments, or enable exploration of higher-order combinatorics that are experimentally intractable. Datasets that are believed to have the appropriate scales are becoming available, such as the Human Cell Atlas,[4] and are collected by projects such as CELLxGENE,[5] which hosts millions of single-cell gene expression profiles from across species, organs, health and disease states.

Two recently published single-cell foundation models, scGPT[6] and scFoundation,[7] were trained on over 33 million and 50 million single cells, respectively. Both are deep learning models built around the transformer architecture.[8] Both are designed for various tasks: cell type annotation, gene network inference, and perturbation prediction. The authors provide the pre-trained models, which an end-user can apply to specific tasks after fine-tuning them with further data sets. For the perturbation prediction, scFoundation forked and modified GEARS,[9] another tool for this task, which uses graph neural networks and shared gene ontology[14] annotations to predict the effect of genetic perturbations.

Results

Here, we benchmark the performance of these three tools for predicting gene expression changes in cell lines upon genetic perturbation.

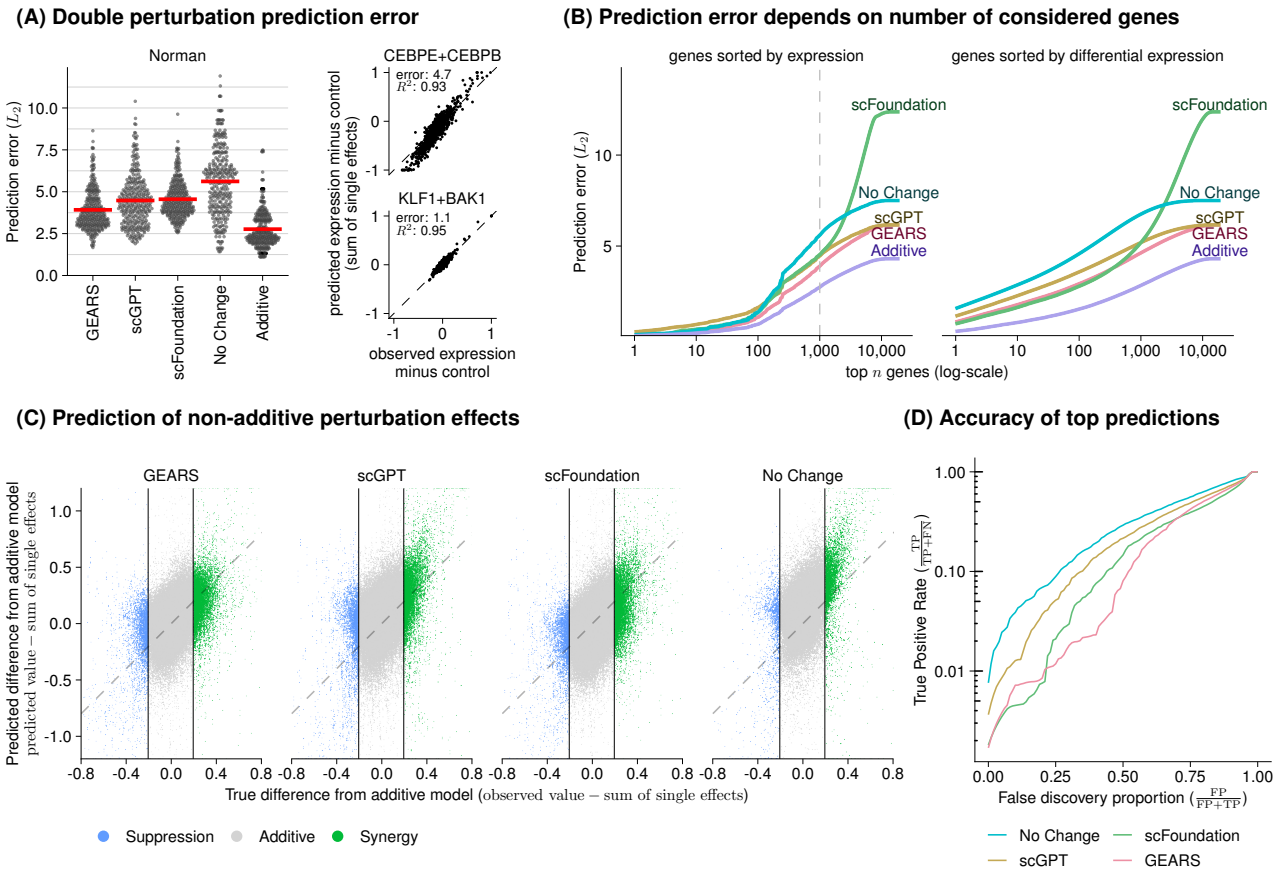


Figure 1 | Double perturbation prediction. (A) Beeswarm plot of the prediction errors for 62 double perturbations and five test-training splits. The prediction error is measured by the L_2 distance between predicted and observed expression profile of the n = 1000 most highly expressed genes. The red horizontal lines indicate the means. The two scatterplots show, exemplarily, observed vs. predicted expression from the additive model for two double perturbations. (B) Prediction error as a function of n, the size of the prediction objective. Left: genes ranked by expression in the control condition, right: by differential expression between observed value and expression in the control condition. The dashed line at n = 1000 is the choice in Panel A and elsewhere in this work. (C) Scatterplot of observed difference from the additive model versus the predicted difference from the additive model. The diagonal dashed line indicates perfect predictions. Each point corresponds to one of the 1000 genes measured by RNAseq under one of the 62 double perturbations. We divided the predictions into three categories, depending on a statistical threshold for the true interaction effect. (D) True positive rate (recall) of the synergistic predictions as a function of the false discovery proportion.

Predicting the effect of unseen double perturbations

We used the dataset of Norman et al. [10], where the expression of 101 individual genes and 124 pairs of genes in K562 cells was, each in turn, upregulated with a CRISPR-based activation system. The phenotypes for these 225 perturbations are the log-normalized RNAseq expression values for 19 264 genes. The data are provided by scFoundation (see Data Availability). We fine-tuned the tools on all 101 single and 62 double perturbations and assessed the prediction error on the remaining 62 double perturbations (held-out data). We ran each analysis five times using different random number generator seeds and thus different held-out sets. For comparison, we considered two baselines: (1) a model that predicts no change, i.e., always the same expression as in the control (unperturbed) condition, and (2) the additive model that, for each double perturbation, predicts the sum of the effects of the two single perturbations. Neither uses any of the double perturbations data.

As shown in Figure 1A, the additive model

outperformed the other methods in terms of prediction error by a large margin. This was also observed when, instead of L_2-distance, we assessed prediction error by the Pearson delta measure (Suppl. Fig. S1).

RNAseq data are noisy, which puts an intrinsic limitation to prediction. Moreover, the data's signal-to-noise ratio is not uniform across genes but depends on the observed count: the lower the count, the larger the signal-to-noise ratio,[11] and the harder the prediction. Hence, we checked how the prediction error depended on that confounding effect. As expected, the prediction error of all methods increased the more low-expressed genes were included in the prediction target (Fig. 1B). However, the ordering of the methods was largely the same across a wide range of prediction target sizes, suggesting that the result of Figure 1A, which uses the top 1000 most highly expressed genes, is representative.

The primary motivation for double perturbation experiments is to find *genetic interactions*, i.e., surprising phenotypes that differ from just combining the phenotypes of the individual perturbations. For quantitative phenotypes such as gene expression, this can be operationalized by looking for cases where the effect of the double perturbation differs from the sum of the effects of the individual perturbations. By definition, the additive model is unable to predict such genetic interactions. We asked whether the tested deep learning models would offer an advantage in identifying such interactions.

To define interactions, we computed, for each of the 1000 read-out genes and 62 held-out double perturbations, the difference between the observed logarithmic fold change and the sum of the individual ones. The values showed a narrow unimodal distribution around zero, with pronounced tails on both sides (Suppl. Fig. S2). This observation justifies using the logarithmic fold change as the quantitative phenotype.[12] We used Efron's empirical null approach,[13] as implemented in the *locfdr* package, to decompose this distribution into a null and alternative component, corresponding to genetic additive and non-additive interactions, respectively (Suppl. Fig. S2). From this, we identified a set of 20 095 and 8 060 observations as "true" synergistic and repressive, respectively, at a false discovery rate of 5%.

We then used the three tools to predict synergistic interactions by sorting each prediction by difference to the additive model. We counted the number of true positives as a function of the fraction of false discoveries among all discoveries (Fig. 1D). The simplistic *no change* baseline method, which always predicts the control condition expression, outperformed the other methods. The same ranking of methods was observed in the receiver-operating characteristic (ROC) and precision-recall curves (Suppl. Fig. S3).

We further investigated the top 100 synergistic predictions from each method and found many reoccuring genes and perturbations (Suppl. Fig. S4A,B). The six most reoccurring genes accounted for 18–71% of the top synergistic predictions. For the top two genes, the correct identification of synergistic effects appeared to be largely a side-product of highly variable observed expression values and mostly constant predictions across perturbations (Suppl. Fig. S4C).

Predicting the effect of unseen single perturbations

An exciting functionality of the three deep learning tools is to predict the effect of a genetic perturbation that was not previously seen. The hope is that they might have learned so much about relationships and interactions between genes (and their products) that they can extrapolate to unseen gene perturbations. GEARS uses shared gene ontology[14] annotations. The foundation models scGPT and scFoundation have learned an embedding of the genes into an abstract mathematical space meant to digest all the biological phenomena they encountered in their training.

To benchmark this functionality, we used a CRISPR interference dataset by Adamson et al. [15] obtained with K562 cells, and two datasets by Replogle et al. [16] obtained with K562 and RPE1 cells. The data are provided by GEARS (details in Suppl. Tab. S1 and Data Availability).

As a baseline, we devised a simple linear model as follows: we assume that we have K-dimensional embedding vectors for read-out genes, given as a matrix $\mathbf{G}$, with one row per read-out gene and K columns, and K-dimensional embedding vectors for the perturbations matrix $\mathbf{P}$, with one row per perturbation and K columns. Given a data matrix $\mathbf{Y}^{\text{train}}$ of gene expression values, with one row per read-out gene and one column per perturbation, the

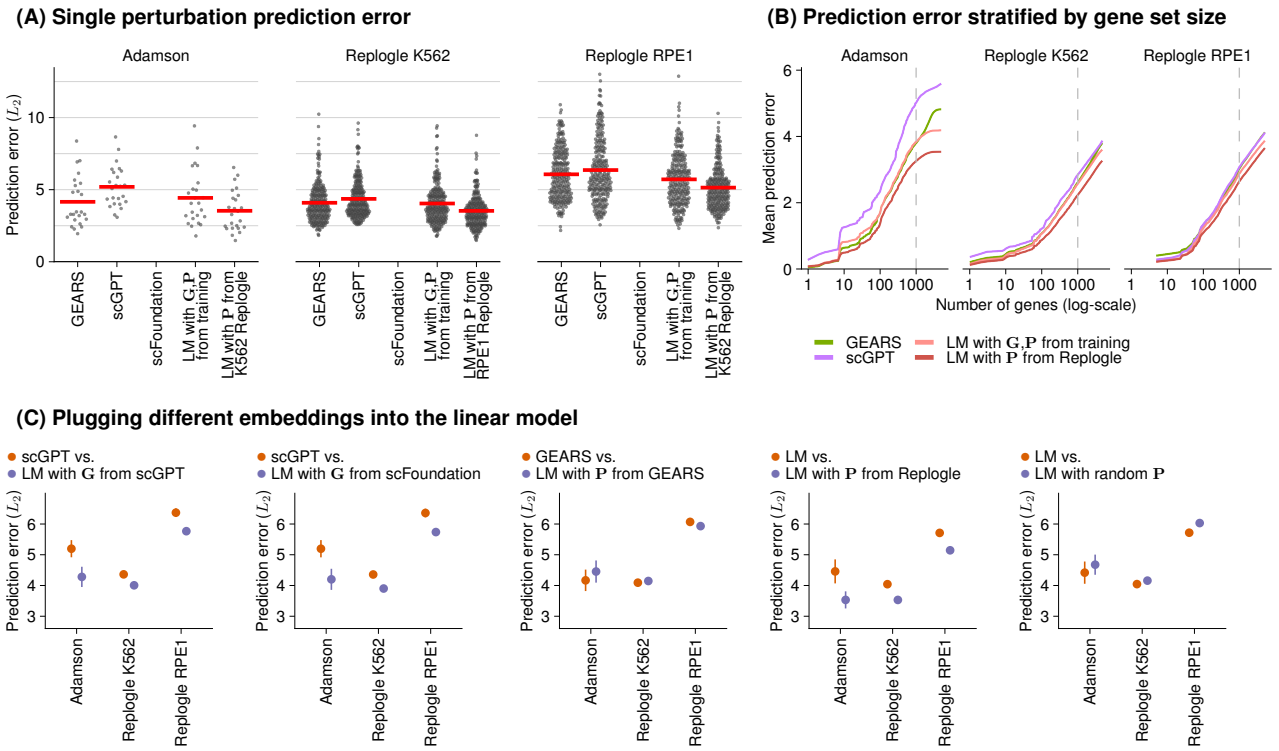


Figure 2 | Single perturbation prediction. (A) Beeswarm plot of the prediction error for 24, 221, and 232 unseen single perturbations of the predicted and observed expression profile for 1 000 most highly expressed genes across three datasets and two test-training splits. The red horizontal line indicates the mean. (B) Line plots of the prediction error stratified by the number of considered genes (sorted by expression). (C) Point-range plots contrasting the performance of different models. The point indicates the mean performance and the line shows the bootstrapped standard error of the mean. *LM*: linear model

model then fits the $K \times K$ matrix $\mathbf{W}$ by minimizing

$$\arg\min_{\mathbf{W}} ||\mathbf{Y}^{\text{train}} - (\mathbf{GWP}^T + \boldsymbol{b})||_2^2, \qquad (1)$$

where $\boldsymbol{b}$ is the vector of average gene expressions. To obtain the embeddings $\mathbf{G}$ and $\mathbf{P}$, we first used the simple approach of performing a PCA on $\mathbf{Y}^{\text{train}}$ and using the top K principal components for $\mathbf{G}$. We then subset this $\mathbf{G}$ to only the rows corresponding to genes that have been perturbed in the training data (and hence appear as columns in $\mathbf{Y}^{\text{train}}$) and used the resulting matrix for $\mathbf{P}$. When using the model for prediction, $\hat{\mathbf{Y}} = \boldsymbol{b} + \mathbf{GWP}^T$, we replace $\mathbf{P}$ with the matrix formed by the rows of $\mathbf{G}$ corresponding to genes perturbed in the test data.

Despite the simplicity of the model and the ad-hoc character of the construction of $\mathbf{P}$, we found that scGPT and GEARS do not outperform it when predicting unseen perturbations (Fig. 2A).

We obtained consistent results when considering more or fewer genes (Fig. 2B) and when using the Pearson delta measure to assess performance (Suppl. Fig. S5). We could not assess scFoundation for this benchmark, as it requires specially preprocessed input data, which we did not have available for the three considered datasets.

Using embeddings from pre-training in a linear model

In the previous section, we constructed the perturbation embedding $\mathbf{P}$ by picking rows from the gene embedding $\mathbf{G}$ rather than from the actual perturbations because the model was not allowed to see the test perturbations in the training phase. Following the idea of transfer learning, we wondered if we could use one data set for pre-training the perturbation embeddings and then learn and test on another one. We found that using the PCA of the perturbations of one of the two Repogle datasets (i.e., of $\mathbf{Y}^T$) as embedding matrix $\mathbf{P}$ improved prediction performance

(Fig. 2A, right-most beeswarms).

Next, we asked whether the embeddings learned by the foundation models offer further value, and thus used for **G** the gene embeddings learned by scGPT and scFoundation during pretraining. The resulting predictions were better than those of scGPT (Fig. 2C). In the same spirit, we converted GEARS' gene ontology annotations into a perturbation embedding **P** by computing a spectral embedding[17,18] of the pathway membership matrix. This improved the predictions on the Replogle et al. [16] RPE1 dataset, but did not make a difference for the other two.

Lastly, we contrasted the prediction performance of the linear model with **G** and **P** from the $\mathbf{Y}^{\text{train}}$ PCA (as done in the previous section) against that of the linear model with **P** from the Replogle et al. [16] data or a random **P** from the standard normal distribution. We found that the informative embedding provided a small but consistent performance benefit (Fig. 2C).

Discussion

We present benchmarks in which current foundation models do not provide a performance benefit over very simplistic linear prediction models, despite significant computational expenses (Suppl. Fig. S6).

Our simple linear models are not intended for making novel predictions. However, as the foundation models benchmarked here could not surpass them, we conclude that the goal of predicting the outcome of not-yet-performed experiments has not been achieved.

This result is not due to rarity of non-additive interactions: we identified 28 155 genes which did not fit the additive null model, at a false discovery rate of 5% (Suppl. Fig. S2B).

The publications that presented the three deep learning methods each included a comparison against different baselines: the authors of GEARS compared it to a fork of CPA,[19] scGPT was compared to GEARS and a linear model, and scFoundation was compared to GEARS. Some of these baselines may have happened to be particularly "easy". For instance, CPA was primarily designed to extrapolate to unseen cell types, not to unseen perturbations, and the linear model used in scGPT's benchmark appears to have been set up such that for unseen perturbations, it reverts to predicting no change over the control condition.

Our results are in line with other benchmarks that assessed the performance of foundation models for single-cell data clustering, cell type annotation, batch correction, or perturbation prediction, which all found no added benefit over simpler approaches.[20,21,22] Our results also concur with a study that compared CPA and GEARS against an additive model for double perturbation prediction.[23]

As a side step, we have also explored whether the embeddings learned by the foundational models have value independent from their transformer module, and found hints that this might be the case.

Our negative results on the foundation models' performance in the prediction task should not be seen as arguing against this line of research. Neural networks are not new in single-cell omics – and autoencoder-based architectures are used widely for tasks such as denoising and interbatch and multimodal integration.[24,25] The progress brought by the transformer architecture and the transfer learning paradigm for many machine learning tasks is real and substantial.

However, the application of transformers to single-cell data is currently based on an analogy of genes as words and cells as sentences;[7,6] yet, such theoretical arguments are not sufficient: In the last two decades of machine learning research, which cumulated in the today's LLM revolution, reliable benchmarks with careful design and scoring were essential in guiding the field in the right direction.[26]

Our work provides a benchmark for prediction tasks, together with a baseline that would need to be surpassed by a tool in order to show that actual non-trivial insights have been transferred from the foundation data. Such guides will be crucial to direct further work in bringing transfer learning to biology.

Data Availability

All datasets used in this manuscript are publicly available:

Dataset	Availability (with link)
Norman et al. [10]	Downloaded from scFoundation (Figshare)
Adamson et al. [15]	Downloaded via GEARS from Harvard dataverse
Replogle et al. [16] K562	Downloaded via GEARS from Harvard dataverse
Replogle et al. [16] RPE1	Downloaded via GEARS from Harvard dataverse

Code Availability

The code to reproduce the analyses presented here and details about the software package versions are available at github.com/const-ae/linear_perturbation_prediction-Paper.

Acknowledgments

We thank an anonymous reviewer of one of our previous manuscripts[27] for the suggestion to compare foundation models against simple linear models.

Funding

This work has been supported by the European Research Council (Synergy Grant DECODE under grant agreement no. 810296) and by the Klaus Tschira Foundation (grant 00.022.2019).

References

[1] George I Gavriilidis et al. "A mini-review on perturbation modelling across single-cell omic modalities". In: *Computational and Structural Biotechnology Journal* (2024).

[2] Artur Szałata et al. "Transformers in single-cell omics: a review and new perspectives". In: *Nature Methods* 21.8 (2024), pp. 1430–1443.

[3] Jennifer E. Rood, Anna Hupalowska, and Aviv Regev. "Toward a foundation model of causal cell and tissue biology with a Perturbation Cell and Tissue Atlas". In: *Cell* 187.17 (Aug. 2024), 4520–4545. ISSN: 0092-8674. DOI: 10.1016/j.cell.2024.07.035. URL: http://dx.doi.org/10.1016/j.cell.2024.07.035.

[4] Aviv Regev et al. "The human cell atlas". In: *elife* 6 (2017), e27041.

[5] CZI Single-Cell Biology et al. "CZ CELLxGENE Discover: A single-cell data platform for scalable exploration, analysis and modeling of aggregated data". In: *BioRxiv* (2023), pp. 2023–10.

[6] Haotian Cui et al. "scGPT: toward building a foundation model for single-cell multi-omics using generative AI". In: *Nature Methods* (2024), pp. 1–11.

[7] Minsheng Hao et al. "Large-scale foundation model on single-cell transcriptomics". In: *Nature Methods* (2024), pp. 1–11.

[8] Ashish Vaswani. "Attention is all you need". In: *arXiv preprint arXiv:1706.03762* (2017).

[9] Yusuf Roohani, Kexin Huang, and Jure Leskovec. "Predicting transcriptional outcomes of novel multigene perturbations with GEARS". In: *Nature Biotechnology* 42.6 (2024), pp. 927–935.

[10] Thomas M Norman et al. "Exploring genetic interaction manifolds constructed from rich single-cell phenotypes". In: *Science* 365.6455 (2019), pp. 786–793.

[11] Mark D. Robinson and Gordon K. Smyth. "Moderated statistical tests for assessing differences in tag abundance". In: *Bioinformatics* 23.21 (2007), pp. 2881–2887. DOI: 10.1093/bioinformatics/btm453.

[12] E. Axelsson et al. "Extracting quantitative genetic interaction phenotypes from matrix combinatorial RNAi". In: *BMC Bioinformatics* 12 (2011), p. 342. DOI: https://doi.org/10.1186/1471-2105-12-342.

[13] Bradley Efron. *Large-scale inference: empirical Bayes methods for estimation, testing, and prediction*. Vol. 1. Cambridge University Press, 2012.

[14] The Gene Ontology Consortium. "The Gene Ontology knowledgebase in 2023". In: *Genetics* 224.1 (2023), iyad031. DOI: 10.1093/genetics/iyad031.

[15] Britt Adamson et al. "A multiplexed single-cell CRISPR screening platform enables systematic dissection of the unfolded protein response". In: *Cell* 167.7 (2016), pp. 1867–1882. DOI: 10.1016/j.cell.2016.11.048.

[16] Joseph M Replogle et al. "Mapping information-rich genotype-phenotype landscapes with genome-scale Perturb-seq". In: *Cell* 185.14 (2022), pp. 2559–2575.

[17] Daniel L Sussman et al. "A consistent adjacency spectral embedding for stochastic blockmodel graphs". In: *Journal of the American Statistical Association* 107.499 (2012), pp. 1119–1128.

[18] Gabor Csardi and Tamas Nepusz. "The igraph software package for complex network research". In: *InterJournal* Complex Systems (2006), p. 1695. URL: https://igraph.org.

[19] Mohammad Lotfollahi et al. "Predicting cellular responses to complex perturbations in high-throughput screens". In: *Molecular Systems Biology* (2023), e11517. DOI: 10.15252/msb.202211517.

[20] Kasia Zofia Kedzierska et al. "Assessing the limits of zero-shot foundation models in single-cell biology". In: *bioRxiv* (2023), pp. 2023–10. DOI: 10.1101/2023.10.16.561085.

[21] Rebecca Boiarsky et al. "A deep dive into single-cell RNA sequencing foundation models". In: *bioRxiv* (2023), pp. 2023–10. DOI: 10.1101/2023.10.19.563100.

[22] Tianyu Liu et al. "Evaluating the Utilities of Foundation Models in Single-cell Data Analysis". In: *bioRxiv* (2024). DOI: 10.1101/2023.09.08.555192.

[23] Thomas Gaudelet et al. "Season combinatorial intervention predictions with Salt & Peper". In: *arXiv preprint arXiv:2404.16907* (2024).

[24] "A Python library for probabilistic analysis of single-cell omics data". In: *Nature Biotechnology* (2022). ISSN: 1546-1696. DOI: 10.1038/s41587-021-01206-w. URL: https://doi.org/10.1038/s41587-021-01206-w.

[25] Malte D Luecken et al. "Benchmarking atlas-level data integration in single-cell genomics". In: *Nature methods* 19.1 (2022), pp. 41–50.

[26] "A Survey on Evaluation of Large Language Models". In: *ACM Transactions on Intelligent Systems and Technology* 15.3 (Mar. 2024), 1–45. ISSN: 2157-6912. DOI: 10.1145/3641289.

[27] Constantin Ahlmann-Eltze and Wolfgang Huber. "Analysis of multi-condition single-cell data with latent embedding multivariate regression". In: *bioRxiv* (2024).

—

Methods

We ran the double perturbation benchmark on the data produced by Norman et al. [10] and reprocessed by scFoundation (details in Data Availability). For the single gene perturbation benchmarks, we used the data from Adamson et al. [15] and Replogle et al. [16] as provided by GEARS (details in Data Availability).

We ran GEARS version 0.1.2, scGPT version 0.2.1, and scFoundation (which is build on top of a GEARS version 0.0.2 fork). We used each tool, as much as possible, with their default parameters. We limited the fine-tuning time to three days, which meant that we were only able to train scFoundation for 5 epochs.

For the double perturbation benchmark, we split the data into test and training sets. We assigned all single gene perturbations and a randomly chosen half of the double perturbations to the training and used the other half of the double perturbations to the test set. To ensure reproducibility, we repeated the test-training splitting five times. For the single perturbation benchmark, we used GEARS *simulation* test-training splitting procedure, which we repeated twice.

For the double perturbation benchmark, we used two baseline models: no change and additive. The *no change* model "predicted" for each

double perturbation the expression values seen in the control condition. The *additive* model predicted the gene expression for a double perturbation of gene A and B

$$\boldsymbol{x}^{\mathrm{AB}} = \boldsymbol{x}^{A} + \boldsymbol{x}^{B} - \boldsymbol{c}, \tag{2}$$

where $\boldsymbol{x}^{A}$ and $\boldsymbol{x}^{B}$ are the observed expression vectors for the single perturbation of gene A and B, respectively, and $\boldsymbol{c}$ is the expression in the control condition.

To predict the effects of unseen single perturbations, we used a simple linear model as described in eq. (1). We set $\boldsymbol{b}$ to the row means of the training data ($\boldsymbol{b} = 1/N \sum_i \mathbf{Y}^{\mathrm{train}}_{:i}$) and find $\mathbf{W}$ using the normal equations

$$\mathbf{W} = (\mathbf{G}^T\mathbf{G} + \lambda\mathbf{I})^{-1}\mathbf{G}^T(\mathbf{Y} - \boldsymbol{b})\mathbf{P}(\mathbf{P}^T\mathbf{P} + \lambda\mathbf{I})^{-1}, \tag{3}$$

where we use a ridge penalty of $\lambda = 0.1$ for numerical stability.

The additive model is a special case of the linear model (1) where the gene embedding is simply the single perturbation data, without any further transformation or dimension reduction ($\mathbf{G} = \mathbf{Y}^{\mathrm{single}}$), the perturbation embedding $\mathbf{P}$ is a binary coding, where each column vector has ones in the rows of the perturbed genes and is zero otherwise, W is the identity matrix, and $\boldsymbol{b} = -\boldsymbol{c}$.

We measured the prediction error using the $L_2(\boldsymbol{x}, \boldsymbol{y}) = \sqrt{\sum_g (x_g - y_g)^2}$ distance between the observed expression values and predictions for the 1 000 most highly expressed genes in the control condition. We also calculated the Pearson delta correlation metric, as suggested by Cui et al. [6]: $\mathrm{PearsonDelta}(\boldsymbol{x}, \boldsymbol{y}) = \mathrm{cor}(\boldsymbol{x}-\boldsymbol{c}, \boldsymbol{y}-\boldsymbol{c})$, where $\boldsymbol{c}$ is the expression in the control condition.

For the double perturbation data, we assess the true positive rate (recall) as a function of the false discovery rate. First, we find the order statistic of the predictions across all test perturbations $\boldsymbol{j} = (j : \mathrm{vec}(\mathbf{X})_j > \mathrm{vec}(\mathbf{X})_{j+1})$, where $\mathbf{X}$ is the matrix of the predictions for all genes and perturbations. The false proportion rate (FDP) at position l for a threshold u, that separates the synergistic from the additive observations, is

$$\mathrm{FDP}_l = \frac{\sum_{i=1}^{l} \mathrm{vec}(\mathbf{Y} - \mathbf{X}^{\mathrm{add}})_{\boldsymbol{j}_i} < u}{l} \tag{4}$$

and the true positive rate as

$$\mathrm{TPR}_l = \frac{\sum_{i=1}^{l} \mathrm{vec}(\mathbf{Y} - \mathbf{X}^{\mathrm{add}})_{\boldsymbol{j}_i} > u}{l}, \tag{5}$$

where $\mathbf{Y}$ is the matrix of observed values and $\mathbf{X}^{\mathrm{add}}$ are the predictions from the additive model. Lastly, we find the order statistic of the false discovery proportions $\boldsymbol{s} = (s : \mathrm{FDP}_s < \mathrm{FDP}_{s+1})$ and plot the tuples $1, 2, \cdots, l$

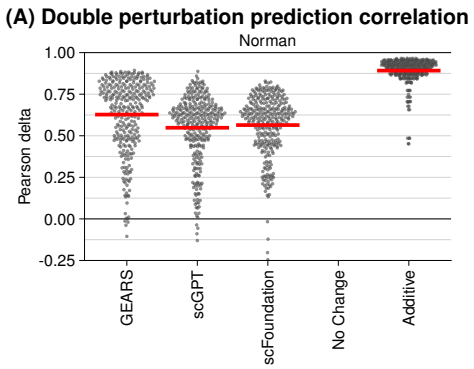
$$(\mathrm{FDP}_{\boldsymbol{s}_l}, \max_i^{1\cdots l}(\mathrm{TPR}_{\boldsymbol{s}_i})). \tag{6}$$

The advantage of false discovery vs. true positive curve over the traditional precision-recall or receiver operator curve is that it provides a direct assessment which fraction of synergistic a method can predict for a fixed fraction of false positives.

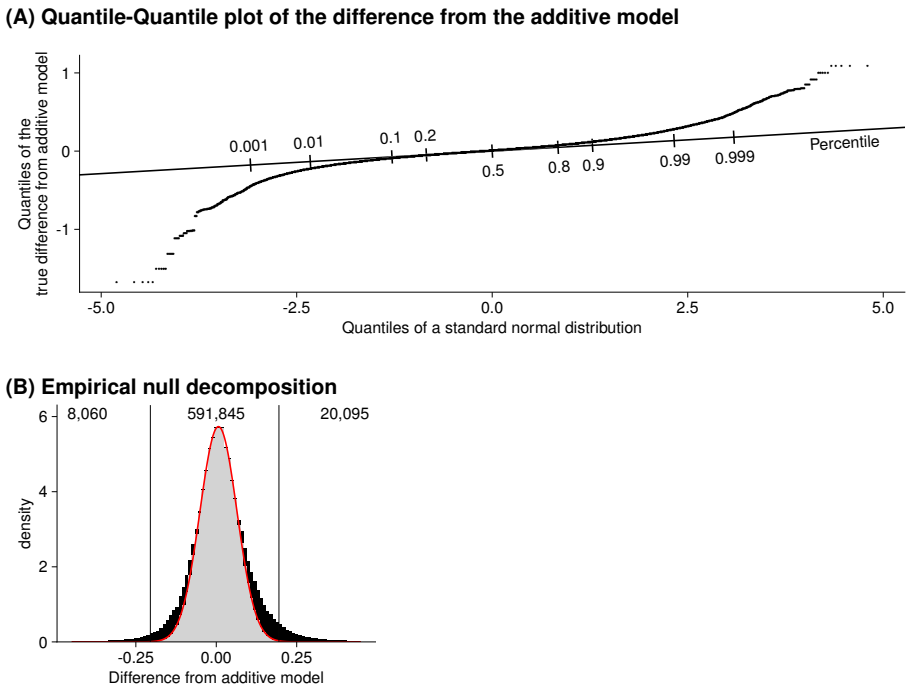
Supplement

Suppl. Table S1 | Overview of the datasets.

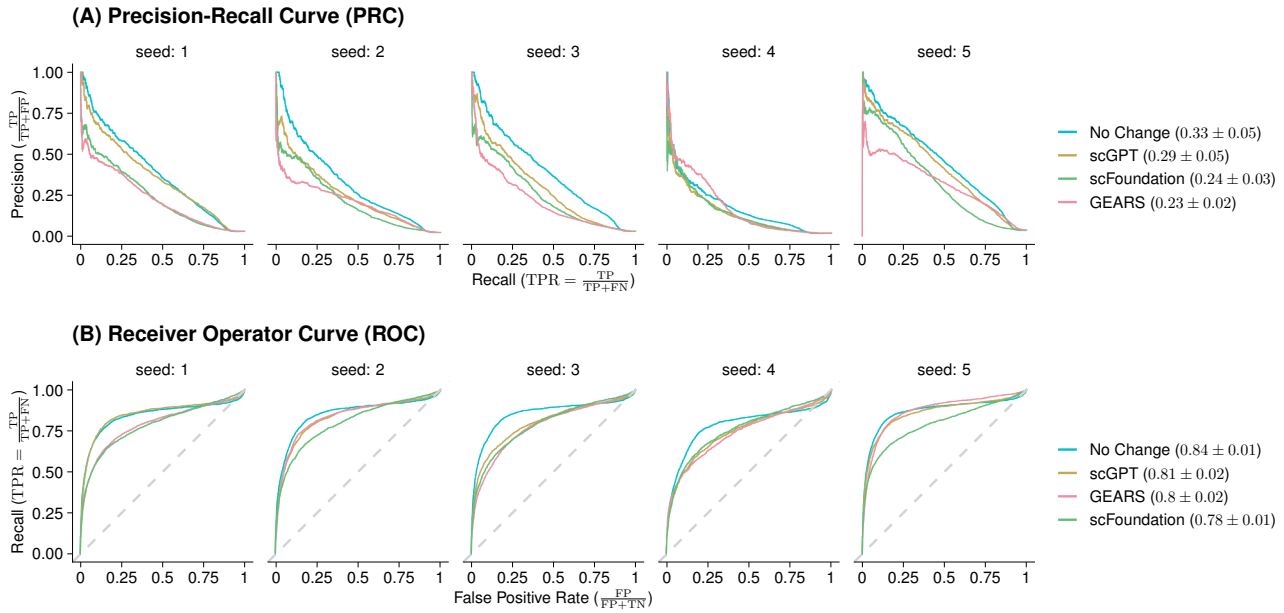
Dataset	Size (Genes × Cells)	Perturbations
Norman et al. [10]	19 264 × 84 143	124 double and 100 single perturbations
Adamson et al. [15]	5 060 × 65 899	81 single perturbations
Replogle et al. [16] K562	5 000 × 162 264	1 088 single perturbations
Replogle et al. [16] RPE1	5 000 × 161 423	1 535 single perturbations



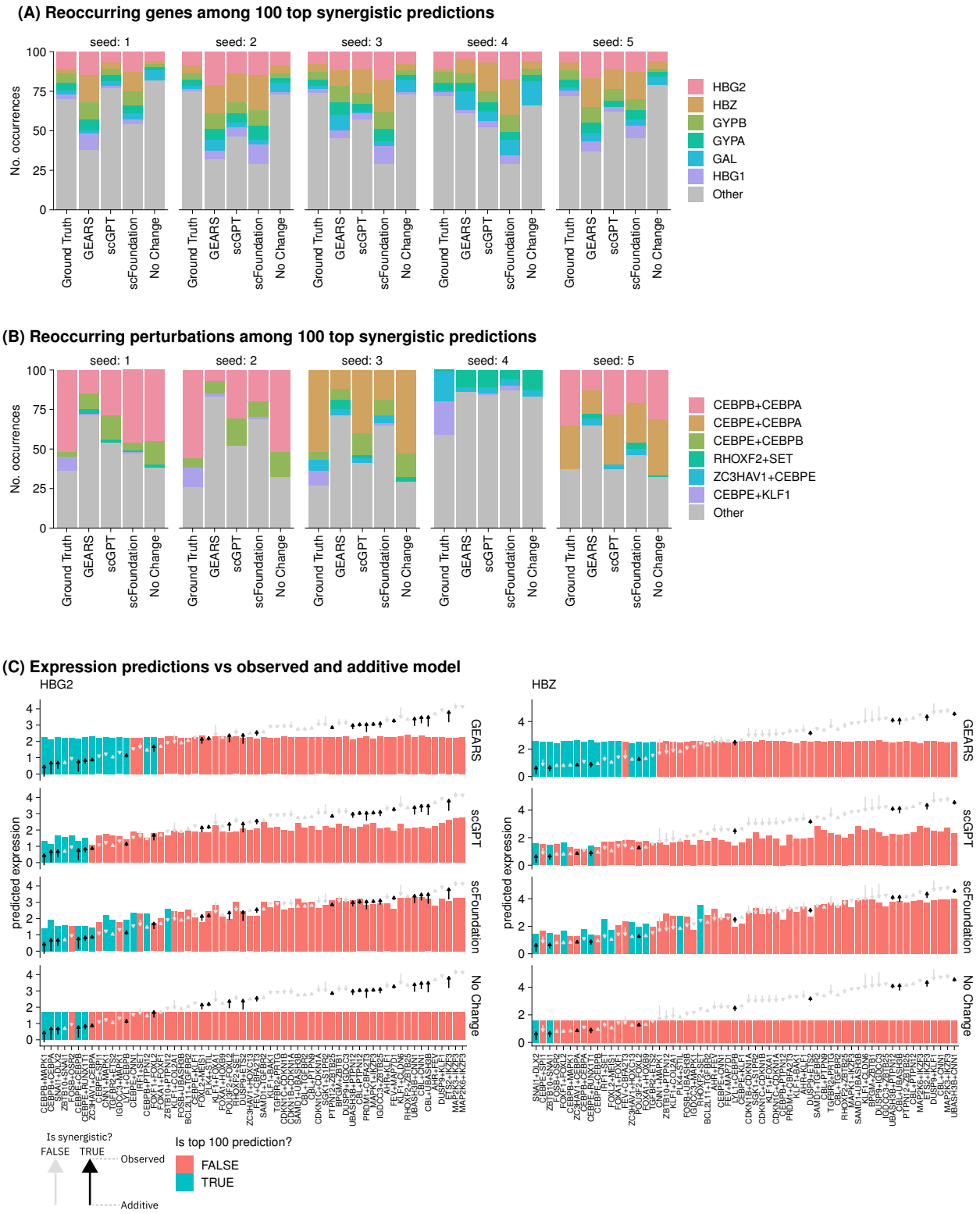
Suppl. Figure S1 | Double perturbation performance measured with Pearson delta. The Pearson delta measure calculates the correlation of the prediction and observations after subtracting the expression in the control condition. The correlation for the *no change* predictions could not be calculated because they were all zero. The horizontal red line shows the mean per condition.



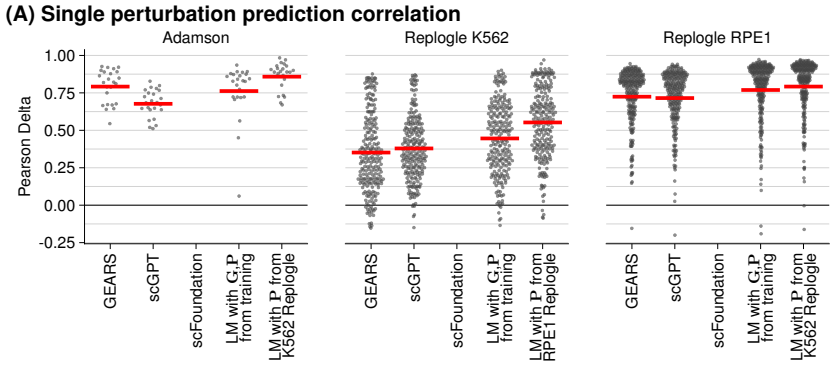
Suppl. Figure S2 | Distribution of the difference from the additive model. (A) Quantile-quantile plot comparing the distribution of the differences between observed gene expression values under double perturbations to the prediction of the additive model against a standard normal distribution. The slope of the line is the standard deviation of the null model. (B) Histogram of the differences with a red curve overlayed that shows the null distribution fitted using *locfdr*. Values under the curve are show in grey and the black bars show the observations that exceed what we would expect under the null model. The vertical bar shows the upper and lower thresholds for which the observations have a false discovery rate of less than 5%. The number count the observations per group.



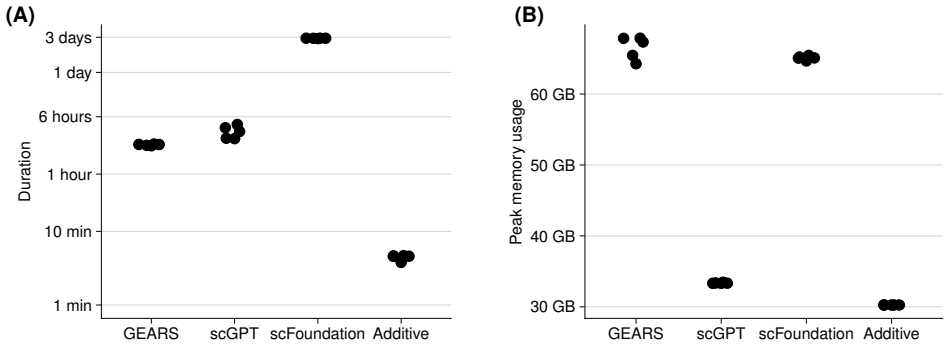
Suppl. Figure S3 | Performance measures of non-additive perturbation effects. (A) Precision-recall and (B) receiver operator curve for four methods of predicting which genes have synergistic perturbation effects. The numbers in parenthesis are the area under the curve (AUC) with the standard error.



Suppl. Figure S4 | Reoccurrence of genes and perturbations for top predictions (A) Reoccurrence of genes and (B) and perturbations among the 100 predictions that differed most from the additive model. The data is facetted by test-training split. The ground truth column shows the genes and perturbations sorted by observed difference from the additive prediction. The highlighted genes and perturbations are the six most reoccurring ones. (C) Comparison of the predicted expression against the predictions from the additive model (base of the arrow) and the observed expression values (arrow tip) for all test perturbations with seed = 1. The arrow color indicates if the difference between observed and additve value exceeded the synergistic threshold. The bars are colored blue if the prediction was among the 100 predictions that differed most from the additive model.



Suppl. Figure S5 | Single perturbation performance measured with Pearson delta. The Pearson delta measure calculates the correlation of the prediction and observations after subtracting the expression in the control condition. The horizontal red line shows the mean per condition.



Suppl. Figure S6 | Computational resource requirements. The resource usage needed for measured for the Norman dataset which had 19 624 genes and 81 143 cells grouped into 225 conditions. Each point is one of the five test-training splits. (A) Elapsed time on a log scale to fine-tune and predict the double perturbations. (B) Peak memory usage for each tool measured with GNU time.