16th Place Solution - Los Rodríguez

Introduction

This document contains our submission for the "Judges prizes" competition of the Open-Problems Single-Cell perturbations challenge [1]. In this competition, the main objective was to predict the effect of drug perturbations on peripheral blood mononuclear cells (PBMCs) from several patient samples. For convenience, we have created a Python package with the model here https://github.com/scapeML/scape.

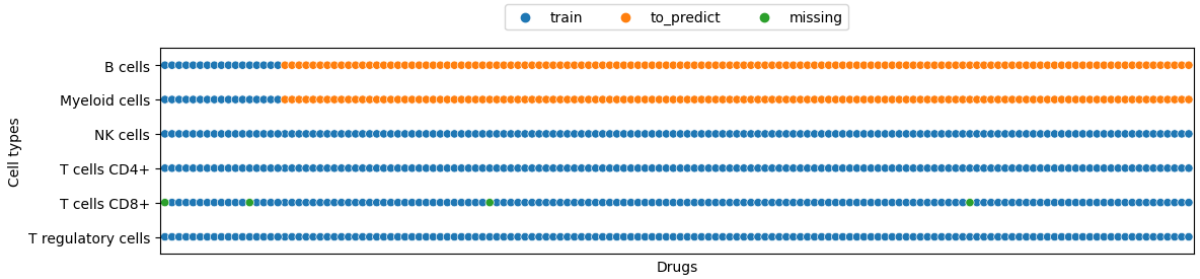

Figure 1. Overview of available data.

Similar to most problems in biological research via omics data, we encountered a high-dimensional feature space (~18k genes) and a low-dimensional observation space (~614 cell/drug combinations) with a low signal-to-noise ratio, where most of the genes show random fluctuations after perturbation. The main data modality to be predicted consisted of signed and log-transformed P-values from differential expression (DE) analysis. In the DE analysis, pseudo-bulk expression profiles from drug-treated cells were compared against the profiles of cells treated with Dimethyl Sulfoxide (DMSO). In addition, challenge organizers also provided the raw data from the single-cell RNA-Seq experiment and from an accompanying ATAC-Seq experiment, conducted only in basal state.

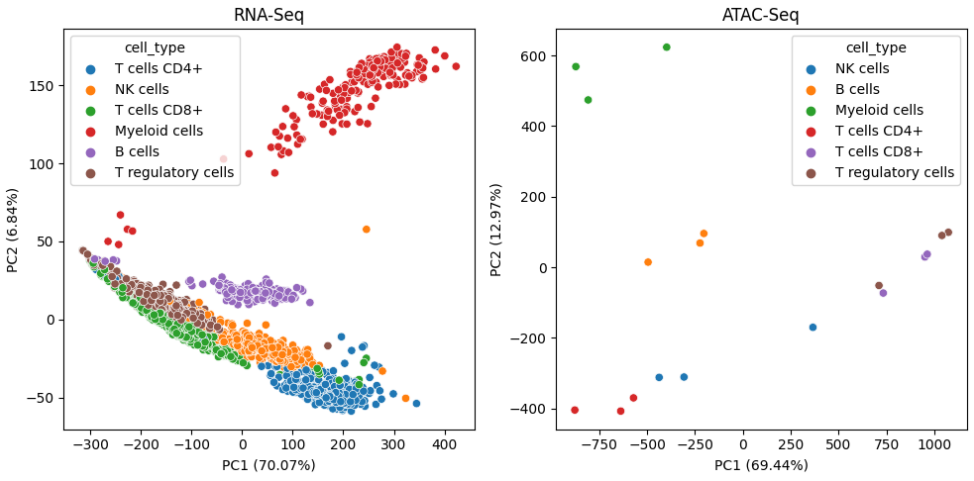

Figure 2. PCA analysis of pseudocount data for RNA-Seq and ATAC-Seq data.

At the beginning of the challenge, we tested different models using the signed log-pvalues ("de_train" data) alone, such as simple linear models, ensembles of gradient boosting with drug and cell features, conditional variational autoencoders, etc. We soon realized that a simple Neural Network using only a small subset of genes to compute drug and cell features (median of the genes grouped by drug and cell) was enough to have a competitive model.

Figure 3 shows the final architecture used for our submission. We used a Neural Network that takes as inputs drug and cell features and produces signed log-pvalues. Features were computed as the median of the signed log-pvalues grouped by drugs and cells, calculated from the de_train.parquet file. Additionally, we also estimated log fold-changes from pseudobulk expression, to produce a matrix of the same shape as the de_train data but containing LFCs instead. We also computed the median per cell/drug as features.

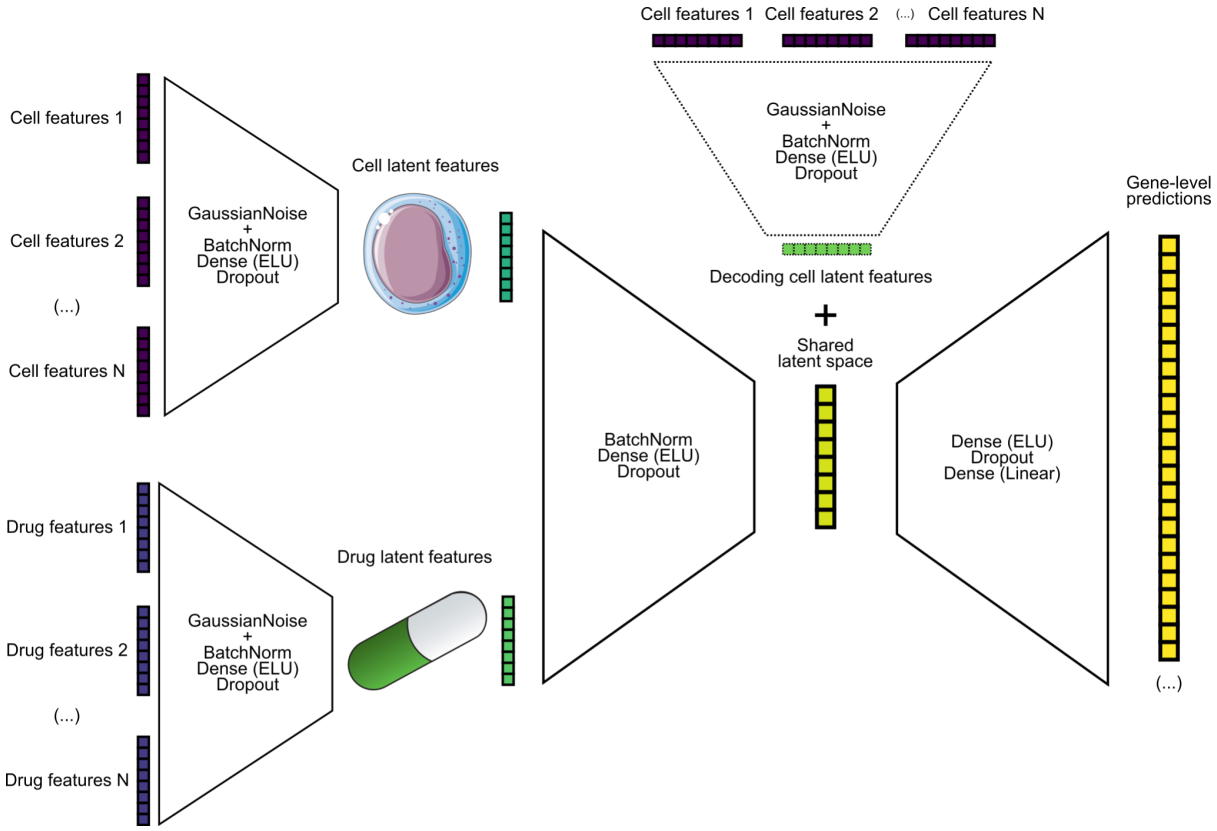

Figure 3. Schematic overview of the model. The additional features used only for the final submission are indicated with a dashed line.

Similar to a Conditional Variational Autoencoder (CVAE), we used cell features both in the encoding part and the decoding part of the NN. Initially, the model consisted of a CVAE that was trained using the cell features as the conditional features to learn an encoding/decoding function conditioned on the particular cell type. However, after testing different ways to train the CVAE (similar to a beta-VAE with different annealing strategies for the KL term), we finally considered a non probabilistic NN since we did not find any practical advantage in this case with respect to a simpler non-probabilistic NN.

Model selection

We tested several CV strategies. To focus on the Judges prizes questions, we decided to adopt a local CV strategy that would resemble as much as possible the competition setup. To this end, we selected the cell type closest to B cells and Myeloid cells in the PCA: **The NK cells**. To mimic as much as possible the real challenge, we also filtered data from B cells and Myeloid cells, which reduced the number of available observations. Finally, we also performed the DE analysis leaving out the local test data, in order to prevent data leakage from the test to the training data. To perform the DE analysis we employed the code provided by the organizers of the challenge [2].

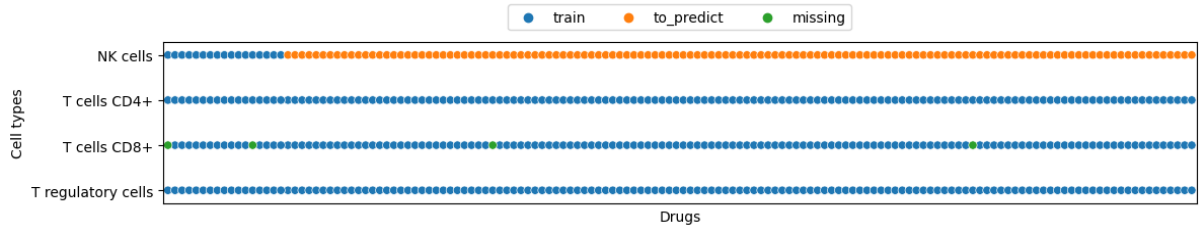

Figure 4. Overview of available data in our local CV setup.

It is important to mention that for our final submissions, we decided to adopt a leave-one-drug-out strategy for all drugs tested in NK-cells. We reasoned that removing 146 observations from the training data would seriously impact the performance of our final model in the leaderboard (LB) evaluation. The other advantage of adopting a leave-one-drug-out approach for NK-cells is that it allows us to estimate how well the model generalizes to unseen drugs, on a per-drug basis per cell type. This approach, combined with two simple baselines (predicting zeros or predicting the median of the log-pvals grouped by drug), provides a more comprehensive understanding of the performance of the models we were training.

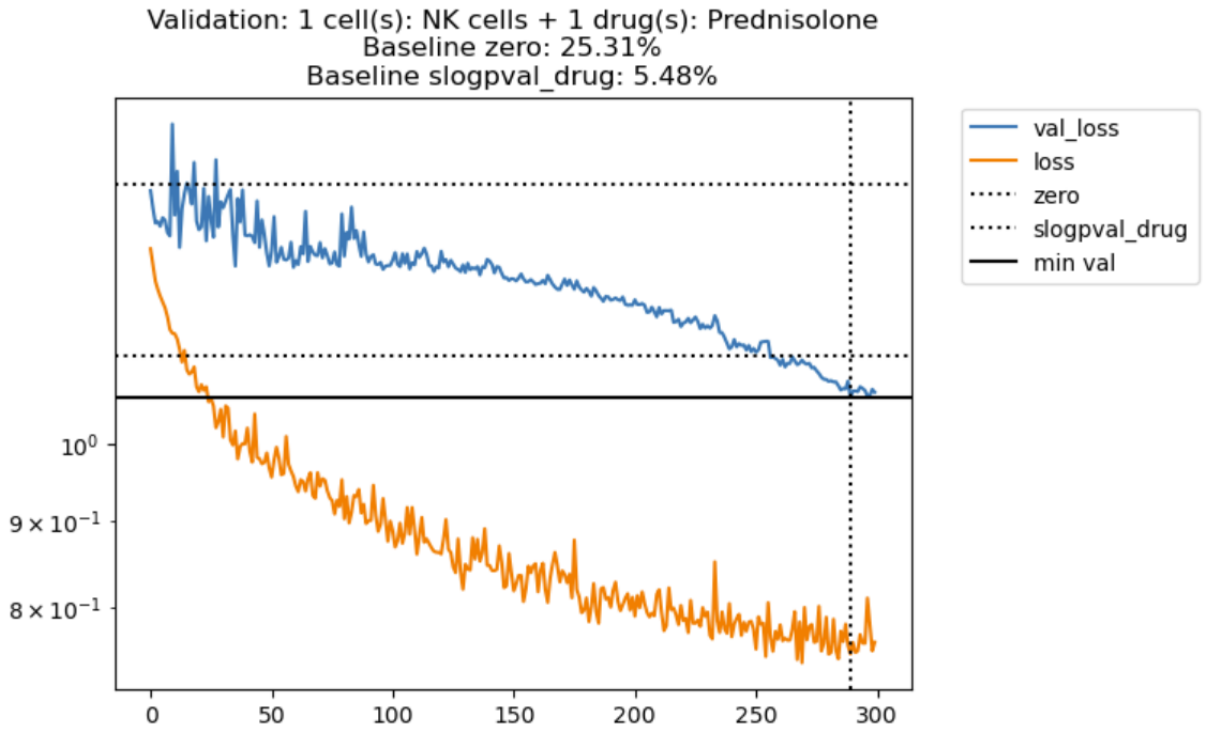

Figure 5. Example of the plots we produced during training. The figure shows the estimated error (MRRMSE) for Prednisolone on NK cells on Y axis, and epochs on X axis. The horizontal dotted lines correspond to the two baselines used to compare the performance (prediction of zeros and prediction of median of p-values on drugs). Positive percentages indicate an improvement over baseline.

At the beginning of the competition, we tested several predictive models, including:

1. Basic inference approaches, including methods for missing values imputation such as median and mean based imputation (both for drug and cells) and imputation using k-nearest neighbors.
2. Regression using Gradient Boosting Trees, which we tested using the libraries XGBoost and CatBoost. We tested using one model per gene, one model per challenge cell type, and one model for the whole dataset.
3. Regression using an autoencoder (AE) neural network architecture implemented with Keras/Tensorflow.
4. Regression using a variational conditional autoencoder (CVAE) neural network architecture, also implemented with Keras/Tensorflow.

For the regression models, we tested different losses, including mean squared error (MSE), mean absolute error (MAE) and the metric used in the challenge, the Mean Rowwise Root Mean Squared Error, (MRRMSE). From the different models and losses that we tested, we found the neural networks and MRRMSE loss to provide the best results, with a public LB score of 0.564 in our first test submissions. This is the model that we used to answer the questions.

It is important to highlight that we designed our autoencoder to have two main components: **A drug component and a cell-type component**. This helped us to test alternative features both for the cell types and drugs, and to evaluate the importance of the different inputs.

In our final submissions, we included an extra component in our model. We used the cell latent features in the decoding layer and noticed better scores in LB by using this approach. Yet, we couldn't see the same score boost in our local CV setup. We think this might be due to having fewer observations to work with. To truly test if the more complicated model improved performance in the challenge, we would have required access to the entire dataset.

Selected submission

For the final submission, we used the NN represented in Figure 3. We trained first the NN on the `de_train` data and the LFCs estimated from pseudobulk gene expression using only the top 64 genes sorted by variance. In total, we generated 146 models (one per drug on NK cells), and we used each model to make the predictions on the challenge drugs/cells. Results were aggregated using the median. This produced what we call our base predictions.

We then trained the same NN model on a subset of the data. This time we selected the top 256 genes (excluding control drugs on the calculation of the variance for the genes) and top 60 drugs. Again, we repeated the same CV strategy of leaving on drug out for NK cells, which resulted in 146 trained models. We repeated the same procedure using the median to generate a smaller set of predictions. This produced what we call enhanced predictions (only for the selected genes and drugs)

Finally, we performed a weighted mean between the base predictions and the enhanced predictions using 0.20/0.80 weights. A full pipeline with the approach using the scape package is available here: https://github.com/scapeML/scape/blob/main/docs/notebooks/solution.ipynb.

Specific questions

1. Integration of biological knowledge

How did you integrate LINCS data? How did this improve your model? Did you use other data sources? Which ones, why?

We decided against using LINCS data in our model because it primarily focuses on cancer cell data, which tends to have a molecular state quite distinct from the PBMCs we were investigating. Despite our exploration of published work and datasets related to predicting

drug-induced changes in single-cell states [3], none of them encompassed the vast array of drug perturbations examined in the challenge. Additionally, we chose not to integrate external data into our approach due to concerns about handling batch effects caused by differences in laboratory settings, protocols, and other related factors.

What representation of the single-cell data did you use? Did you reduce genes into modules?

In terms of features, we tested:

1. Dummy binary variables for cell types and drugs.
2. Basal omics features, including average expression in DMSO and average accessibility per the ATAC-Seq data.
3. Summary statistics of the drug response after grouping by cell type and drug, including standard deviation, mean and median.
4. A “raw” fold-change computed over the raw counts of the single-cell RNA-Seq data (this is, without the corrections applied by limma).
5. Centroids of the principal component space of the drug response data, using cell-type and drug as grouping variables.

And we obtained the best results using the median of the response after grouping by cell type and drug in combination with the raw fold changes, using only a subset with the most variable genes in the dataset.

How did you integrate the ATAC data? Which representation did you use?

To test whether the basal ATAC-Seq data would enhance our predictions, we reasoned that it could be added to the cell-type component of our model. To include the basal ATAC-Seq data, we used the average accessibility of the top 128 most variable genes, with the hypothesis that those are more informative about different cell-type molecular identities. Specifically, we employed average log transformed pseudobulk counts from the ATAC-Seq data as additional cell features in our model.

However, our analysis revealed that incorporating ATAC-Seq features did not enhance the predictive accuracy of our model. We found that the most informative features for cellular behavior are the observed p-values associated with a range of tested drugs. This is particularly evident in the case of B and Myeloid cells, where the best information we can use comes from the limited set of drugs we have on the training data. Features that do not directly come from the perturbational profiles generally fail to contribute additional meaningful data. In some instances, they have even proven to be challenging to implement effectively in our predictive framework. Consequently, there is a significant trade-off to consider: the effort and resources required to integrate and analyze these additional features may not justify the marginal gains in predictive performance. This suggests a strategic focus on refining and deepening our analysis of drug-related p-values to enhance the predictive capabilities of our model.

Did you use the chemical structures in your model?

Following a similar philosophy as the one we used for ATAC-Seq data, we tested the inclusion of chemical features as part of the drug component of our model. In particular, we evaluated using Morgan fingerprints, which are binary vectors that encode the presence of substructures in a molecule. We used the Morgan fingerprints as implemented in the RDKit library, with a radius of 2 and a number of bits of 512.

However, for the final models, we did not include any information from the chemical structures since it did not lead to better predictions.

Did you learn a gene regulatory network?

We acquired estimates of transcription factor (TF) activity using DecoupleR analysis alongside CollecTRI regulons [4,5]. Unlike ATAC-Seq and Morgan fingerprints, which provided data independent of the drug-response data, TF activities were derived through a linear transformation of the original drug response space, leveraging existing knowledge. Consequently, we utilized TF-level data as a replacement for gene-level input features to assess whether integrating prior knowledge would enhance our predictive capabilities. This is an indirect way of using prior knowledge to derive features for the model. Again, this approach did not lead to better outcomes.

Even though learning a Gene Regulatory Network (GRN) holds promise, particularly for understanding causal mechanisms, there are significant challenges that led us to reconsider their use in this context. Firstly, despite the theoretical potential of GRNs to offer better causal insights for predicting perturbations, there is no straightforward method for their development or for assessing their quality. Secondly, while GRNs could potentially enhance interpretability (assuming the model could be accurately identified), we do not anticipate they would significantly improve results in this specific challenge. The complexities and uncertainties in developing and evaluating GRNs outweigh the potential gains in interpretability and causal understanding, which do not directly translate to improved performance in this particular challenge.

If adding a particular biological prior didn’t work, how did you judge this and why do you think this failed?

We checked our local CV errors and the public LB errors to see if adding more data types and using prior knowledge helped our predictive model. In Figure 5, you can see the local CV errors for the three main sources of chemical/biological information we mentioned earlier. We compared all of them with the basic model, which used the median of the response data and raw fold changes after grouping by cell-type and drug, either separately or combined.

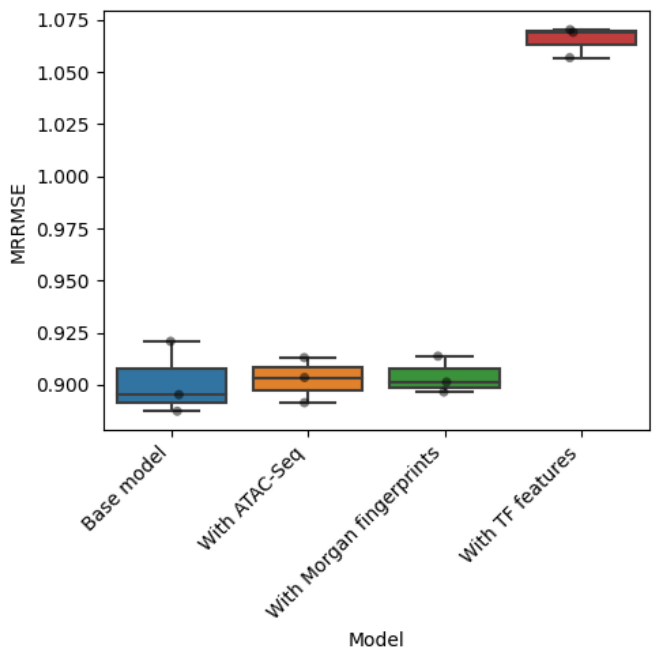

Figure 5. MRRMSE of alternative models in the local CV setup.

Simply put, based on the local CV scores, none of our attempts demonstrated a noteworthy performance improvement compared to the more basic model. The inclusion of GRN features derived from prior knowledge decreased our predictive performance.

2. Exploration of the problem

Are there some cell types it's easier to predict across? What about sets of genes?

To answer this question, we adopted three perspectives: The drug perspective, the cell-type perspective and the gene perspective. These perspectives represent the primary sources of complexity within the dataset. Focusing on the drug-centric viewpoint, we used our local CV strategy to identify specific drugs where our error accumulated.

We found that the error distribution for the drugs was more or less even except for the first four drugs, which accounted for 15% of the total error. As expected, we also found out that the response of drugs that were harder to predict was very different from training cell-types in comparison to the test cell-type. For instance, the drug that accounted for the maximum proportion of the error (IN1451), produced a strong response in NK cells, but seemed to have little effect in T cells CD4+, T cells CD8+ and T regulatory cells.

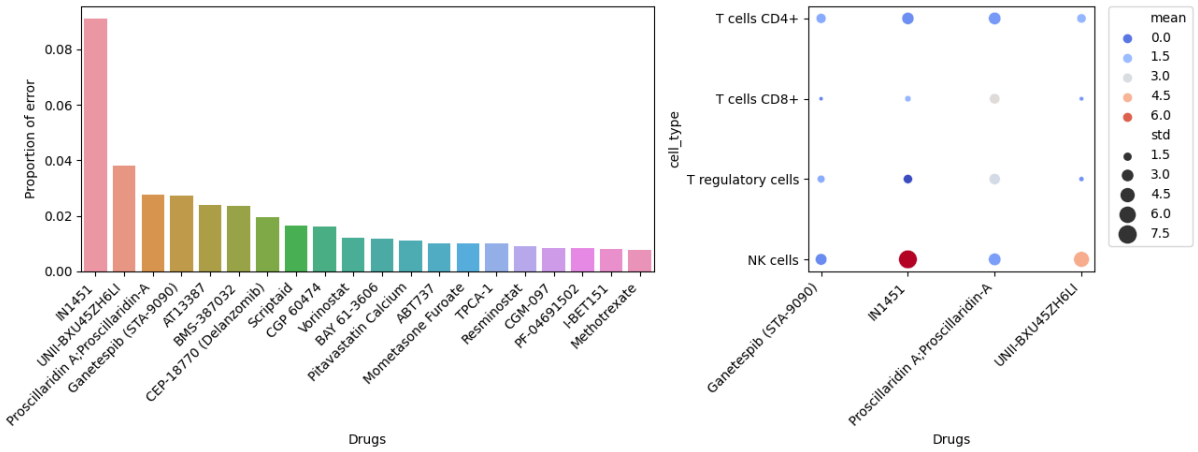

Figure 6. Left panel: Average error for drugs in our local CV setup. Top 20 drugs with highest errors are shown. Right panel: Summary statistics of the drugs with highest errors in the original data.

To better understand cell-type errors, we adjusted our local assessment approach. Our aim was to identify the most challenging cell-type to predict accurately, so we sought a fair representation of each cell-type based on observations. We focused on 15 drugs measured across all cell types and randomly selected 4 for testing. These 4 drugs formed our test set for cell-type cross-validation. In each iteration, the model trained on data from all 15 drugs across different cell-types, excluding the 4 test drugs within a specific cell-type. This method allowed us to evaluate our predictive performance for each cell-type. Figure 7 illustrates the resulting estimates, revealing that myeloid cells were tougher to predict compared to other cell types. This aligns with RNA-Seq PCA analysis findings, indicating the distinctiveness of myeloid cells from the rest, making them more challenging to predict.

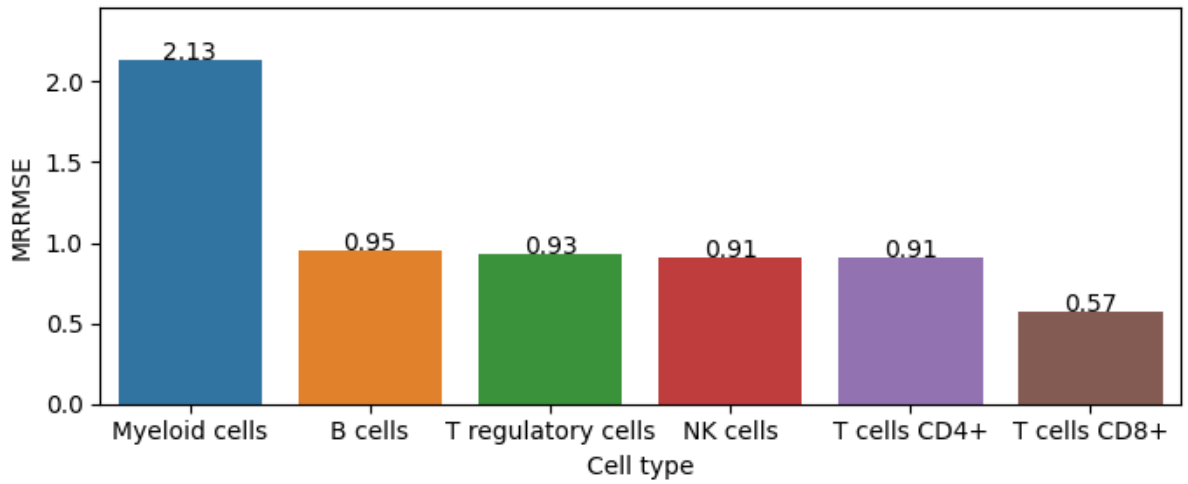

Figure 7. Error estimates for different cell lines.

In terms of genes, we decided to evaluate whether a specific biological function was more difficult to predict than others. To do so, we decided to perform an enrichment analysis of the

top 5% genes with the highest average error in our local CV setup. We performed the analysis using the gene sets from MSigDB hallmarks and decoupleR [4,6]. We found that certain hallmarks like epithelial mesenchymal transition and TNF alpha signaling accumulated a significant amount of genes with high error.

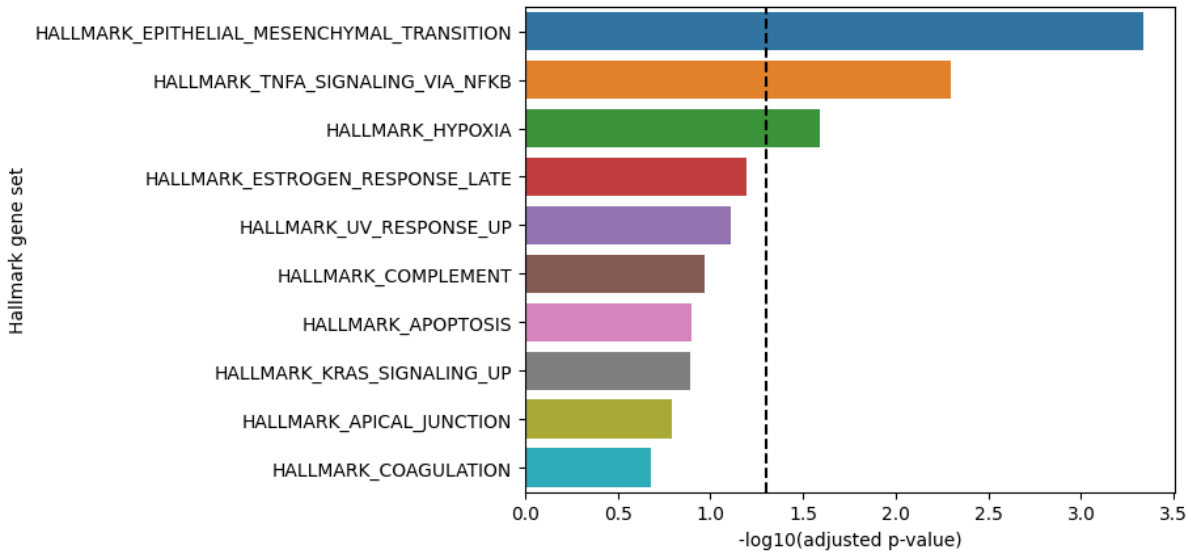

Figure 8. Over-representation analysis results. Bars represent top 10 enriched pathways, and the X axis indicates the -log10 adjusted P value.

Do you have any evidence to suggest how you might develop an ideal training set for cell type translation beyond random sampling of compounds in cell types?

Overall, we found that **unseen and strong** drug responses were the hardest to predict, as per our local CV approach. We believe that our model mostly learned to predict values close to zero, in contrast to the values that represent a strong and novel response in terms of differential expression. We believe that to develop an ideal training set, the key would be to properly cover the gene space in terms of responses. For instance, one could use a pathway enrichment analysis to identify drugs that produce different biological responses, and try to put together a collection of drugs that cover different pathways.

What is the relationship between the number of compounds measured in the held-out cell types and model performance? Is there a sweet spot?

We do not believe that the number of compounds is the key to get a good model performance, but rather an observational space containing responses as diverse as possible, as explained in the previous answer. Of course, a larger number of observations would make training much simpler, and would allow the use of more sophisticated architectures, but we think that the most critical aspect for developing good training is the diversity of observations.

However, we think that a smart split of drugs across different folds could be used to train different models that are diverse enough to combine in ensembles.

3. Model design

Is there certain technical innovation in your model that you believe represents a step-change in the field?

Although we wouldn't label it as a groundbreaking shift in the field, we think our design holds appeal both in machine learning and biology. Our model, involving drug and cell-type components, enables testing novel biological aspects and assessing their impact on model performance, as demonstrated earlier. We anticipate that our model could rapidly integrate new cell and drug features, offering insights into which directions might be more promising to explore.

Can you show that top performing methods can be well approximated by a simpler model?

We wanted to check if simpler models could perform just as well. So, we cut down the input features in our models. Considering that our architecture was simple already, we aimed to find the fewest input features that could match the local CV performance of using the top 128 genes with the highest variance.

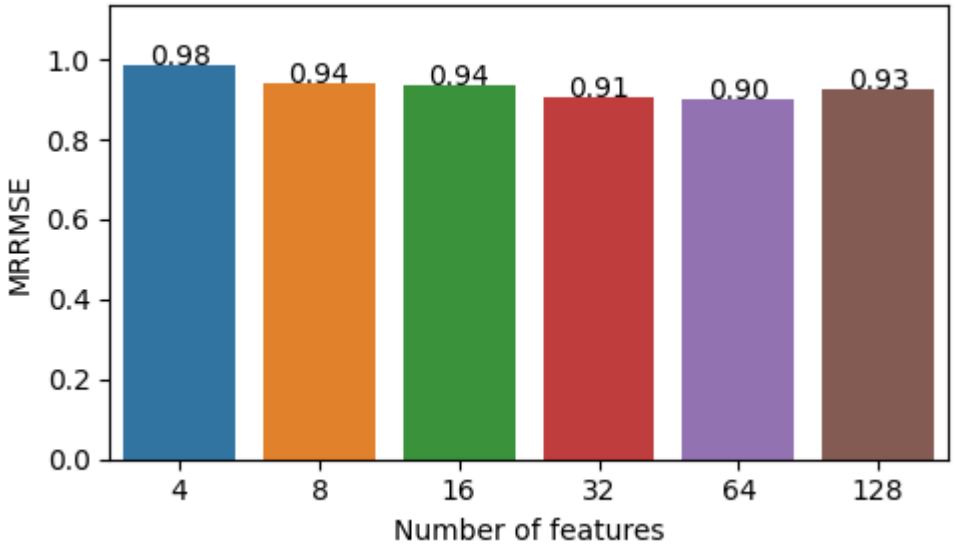

Figure 9. Error with different number of input features.

Interestingly, we found that models with 8 to 64 input features would achieve similar performance that the model that employed 128 features.

Is your model explainable? How well can you identify what is causing your model to respond to certain inputs?

The simplicity of our model allowed us to evaluate which of the input components was most critical for its performance. To do so, we shuffled the order of either the cell or drug input features at the moment of prediction, getting a direct estimate of the relevance of each component.

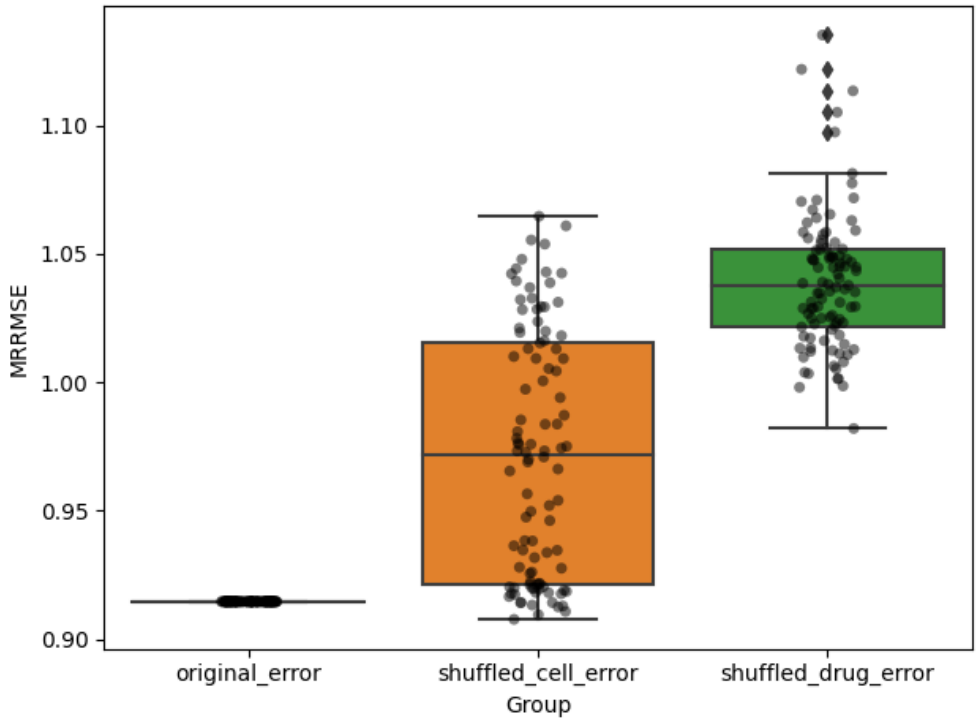

Fig 10. Error distribution after shuffling the cell features or the drug features.

We discovered that while both components had a direct impact on the performance of our model, the mean of the errors after drug features permutation was higher compared to the average error after cell features permutation. This is something to expect, as we have more data points of gene values per drug (146 data points per cell type except B/Myeloid), but we only have 6 data points grouping by cell type. We used this type of permutation tests to estimate the importance that different features had in the CV error.

4. Robustness

Take subsets of the training data (e.g. 95%, 90%, …, 10%). How well does your model perform as a function of percentage of the training data?

This was answered before. Additional experiments on different subset of genes are available at https://github.com/scapeML/scape/blob/main/docs/notebooks/subset-genes.ipynb.

Add small amounts of noise to the input data. What kinds of noise is your model invariant to? Bonus points if the noise is biologically motivated.

To answer this question, we leveraged the Gaussian Noise layer of our model, testing performance across different levels of input noise.

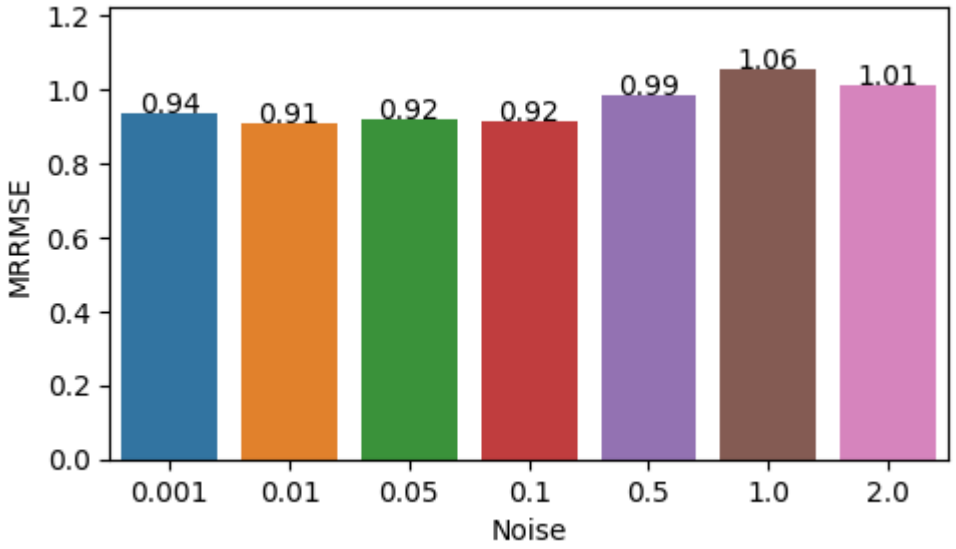

Figure 11. Error distribution with different levels of input Gaussian noise.

We found that an input Gaussian Noise with an standard deviation above 0.1 would be counterproductive for the performance of our model. We did not test additional types of noises.

5. Documentation & code style

For convenience, we refactored the code and created a package called “scape” (https://github.com/scapeML/scape) using https://pdm-project.org/, which contains the code that we finally used for the submission. The code is documented using numpydoc docstrings, and we included a series of notebooks using the *scape* package to learn how to

use it and how to manually create the setup for generating our submission. We have put effort into developing a library that allows for the comfortable configuration and parameterization of neural networks, with an automatic mode for calculating diverse features from drugs and cell lines.

6. Reproducibility

In order to improve reproducibility, we show how the tool package can be installed and used directly from Google Colab https://colab.research.google.com/drive/1-o_IT-ttoKS-nbozj2RQusGoi-vm0-XL?usp=sharing.

We also included an environment.yml file to exactly recreate the environment we used for testing using conda.

References

1. Open Problems – Single-Cell Perturbations. https://kaggle.com/competitions/open-problems-single-cell-perturbations.
2. neurips-2023-scripts/compute_de.ipynb at main · openproblems-bio/neurips-2023-scripts. *GitHub* https://github.com/openproblems-bio/neurips-2023-scripts/blob/main/compute_de.ipynb.
3. Lotfollahi, M. *et al.* Predicting cellular responses to complex perturbations in high-throughput screens. *Mol. Syst. Biol.* **19**, e11517 (2023).
4. Badia-I-Mompel, P. *et al.* decoupleR: ensemble of computational methods to infer biological activities from omics data. *Bioinform Adv* **2**, vbac016 (2022).
5. Müller-Dott, S. *et al.* Expanding the coverage of regulons from high-confidence prior knowledge for accurate estimation of transcription factor activities. *Nucleic Acids Res.* **51**, 10934–10949 (2023).
6. Liberzon, A. *et al.* The Molecular Signatures Database (MSigDB) hallmark gene set collection. *Cell systems* **1**, 417 (2015).