Predicting Single-Cell Perturbation Responses for Unseen Drugs

Anonymous Author(s)
Affiliation
Address
email

Abstract

Single-cell transcriptomics enabled the study of cellular heterogeneity in response to perturbations at the resolution of individual cells. However, scaling high-throughput screens (HTSs) to measure cellular responses for many drugs remains a challenge due to technical limitations and, more importantly, the cost of such multiplexed experiments. Thus, transferring information from routinely performed bulk RNA HTS is required to enrich single-cell data meaningfully. We introduce a new encoder-decoder architecture to study the perturbational effects of unseen drugs. We combine the model with an architecture surgery for transfer learning and demonstrate how training on existing bulk RNA HTS datasets can improve generalisation performance. Better generalisation reduces the need for extensive and costly screens at single-cell resolution. We envision that our proposed method will facilitate more efficient experiment designs through its ability to generate in-silico hypotheses, ultimately accelerating drug discovery.

1 Introduction

Recent advances in single-cell methods allowed the simultaneous analysis of millions of cells, increasing depth and resolution to explore cellular heterogeneity (Sikkema et al., 2022; Han et al., 2020). With single-cell RNA sequencing (scRNA-seq) and high-throughput screens (HTSs) one can now study the impact of different perturbations, i.e., drug-dosage combinations, on the transcriptome at cellular resolution (Yofe et al., 2020; Norman et al., 2019). Unlike conventional HTSs, scRNA-seq HTSs can identify subtle changes in gene expression and cellular heterogeneity, constituting a cornerstone for pharmaceutics and drug discovery (Srivatsan et al., 2020). Nevertheless, these newly introduced sample multiplexing techniques (McGinnis et al., 2019; Stoeckius et al., 2018; Gehring et al., 2018) require expensive library preparation and do not scale to screen thousands of distinct molecules. Even in its most cost-effective version, nuclear hashing, the acquired datasets contain no more than 200 different drugs (Srivatsan et al., 2020).

Consequently, computational methods are required to address the limited exploration power of existing experimental methods and discover promising therapeutic drug candidates. Suitable methods should predict the response to unobserved (combinations of) perturbations. Increasing in difficulty, such tasks may include inter- and extrapolation of dosage values, the generalisation to unobserved (combinations of) drug-covariates (e.g., cell-type), or predictions for unseen drugs. In terms of medical impact, the prediction of unobserved perturbations may be the most desirable, for example for drug repurposing. At the same time, it requires the model to properly capture complex chemical interactions within multiple distinct cellular contexts. Such generalisation capabilities can not yet be learned from single-cell HTSs alone, as they supposedly do not cover the required breadth of chemical interactions. In this work, we leverage information across datasets to alleviate this issue.

Submitted to 36th Conference on Neural Information Processing Systems (NeurIPS 2022). Do not distribute.

We propose a new model that generalises previous work on Fader Networks by Lample et al. (2017) and the Compositional Perturbation Autoencoder (CPA) by Lotfollahi et al. (2021) to the challenging scenario of generating counterfactual predictions for unseen compounds. Our method is as flexible and interpretable as CPA but further enables us to leverage lower resolution but higher throughput assays, such as bulk RNA HTSs, to improve the model's generalisation performance on single-cell data (Amodio et al., 2021). Our main contributions are:

1. We introduce chemCPA, a model that incorporates knowledge about the compounds' structure, enabling the prediction of drug perturbations at a single-cell level from molecular representations.

2. We propose and evaluate a transfer learning scheme to leverage HTS bulk RNA-seq data in the setting of both identical and different gene sets between the source (bulk) and target (single-cell) datasets.

3. We show how chemCPA outperforms existing methods on the task of predicting unobserved drug-covariate combinations. At the same time, we demonstrate chemCPA's versatility and evaluate chemCPA on generalisation tasks that cannot be modeled using any previously existing method.

2 Related Work

Over the past years, deep learning (DL) has become an essential tool for analysis (Lopez et al., 2020) and interpretation (Rybakov et al., 2020) of scRNA-seq data. Representation learning in particular, has been useful not only for identifying cellular heterogeneity and integration (Gayoso et al., 2022), or mapping query to reference datasets (Lotfollahi et al., 2022), but also in the context of modelling single-cell perturbation responses (Rampášek et al., 2019; Seninge et al., 2021; Lotfollahi et al., 2019; Ji et al., 2021).

Unlike linear models (Dixit et al., 2016; Kamimoto et al., 2020) or mechanistic approaches (Fröhlich et al., 2018; Yuan et al., 2021), DL is suited to capture non-linear cell-type-specific responses and easily scales to genome-wide measurements. Recently, Lotfollahi et al. (2021) introduced the CPA method for modelling perturbations on scRNA-seq data. CPA does not generalise to unseen compounds, hindering its application to virtual screening of drugs not yet measured via scRNA-seq data, which is required for effective drug discovery.

For bulk RNA data, on the other hand, several methods have been proposed to predict gene expression profiles for de novo chemicals (Pham et al., 2021; Zhu et al., 2021; Umarov et al., 2021). Crucially, the L1000 dataset, introduced by the LINCS programme (Subramanian et al., 2017), greatly facilitated such advances on phenotype-based compound screening. However, it remains unclear how to translate these approaches to single-cell datasets that include significantly fewer compounds and, in many cases, rely on different gene sets.

3 Chemical Compositional Perturbation Autoencoder

We consider a dataset $\mathcal{D} = \{(x_i, y_i)\}_{i=1}^N = \{(x_i, (d_i, s_i, c_i))\}_{i=1}^N$, where $x_i \in \mathbb{R}^n$ describes the n-dimensional gene expression and y_i an attribute set. For scRNA-seq perturbation data, we usually consider the drug and dosage attributes, $d_i \in \{\text{drugs in } \mathcal{D}\}$ and $s_i \in \mathbb{R}$, respectively, and the cell-line c_i of cell i. Note that this set of attributes $\mathcal{Y}$ depends on the available data and could be extended to covariates such as patient, or species.

A possible approach to predicting counterfactual combinations is to encode a cell's gene expression x_i invariantly from its attributes y_i as a latent vector z_i, called the basal state. Afterwards, z_i can be combined with $z_{\text{attribute}}$ to encode any attribute combination $y_i' \neq y_i$, and decoded back to a gene expression state $\hat{x}_i$ that corresponds to this new set of chosen attributes. To this end, we divide our model, the Chemical Compositional Perturbation Autoencoder (chemCPA), into three parts: (1) the gene expression encoder and decoder, (2) the attribute embedders, and (3) the adversarial classifiers, see Figure 1 for an illustration.

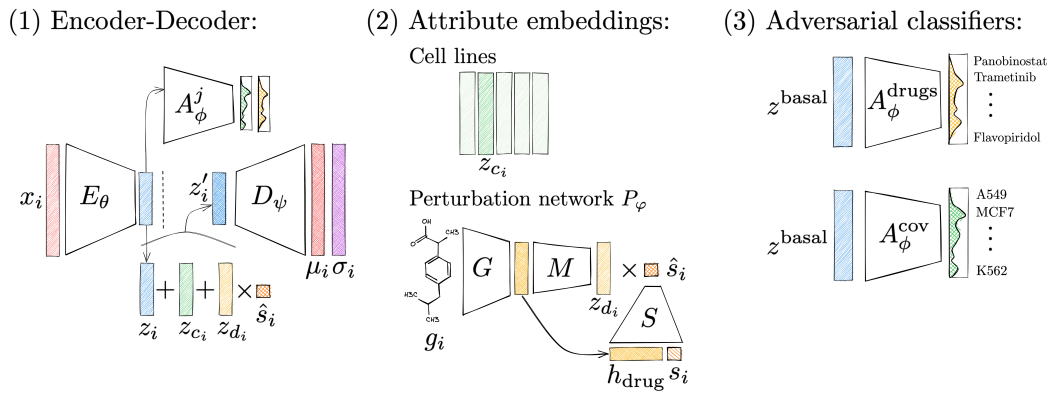

Figure 1: Architecture of chemCPA. The model consists of three parts: (1) the encoder-decoder architecure, (2) the attribute embeddings, and (3) the adversarial classifiers. The molecule encoder G can be any graph- or language-based model as long as it generates fixed-sized embeddings h_{drugs}. The MLPs S and M are trained to map the embeddings to the perturbational latent space. There, z_{d_i} is added to the basal state z_i and the covariate embedding z_{c_i}. In this work, the latter always corresponds to cell lines. The basal state $z_i = E_\theta(x_i)$ is trained to be invariant through adversarial classifiers A^j_ϕ and the decoder D_ψ gives rise to the Gaussian likelihood $\mathcal{N}(x_i \,|\, \mu_i, \sigma_i)$.

3.1 Gene encoder and decoder

Following Lotfollahi et al. (2021), our model is based on an encoder-decoder architecture combined with adversarial training. The encoding network $E_\theta : \mathbb{R}^n \to \mathbb{R}^l$ is a multi-layer perceptron (MLP) with parameters θ that maps a measured gene expression state $x_i \in \mathbb{R}^n$ to its l-dimensional latent vector $z_i = E_\theta(x_i)$. Through adversarial classifiers, z_i is trained to not contain any information about its attributes y_i. This gives us control over the latent space in which we update z_i with an additive attribute embedding of our choice and obtain z'_i.

The decoder $D_\psi : \mathbb{R}^l \to \mathbb{R}^{2n}$ is an MLP that takes z'_i as input and computes the component-wise parameters of the underlying distribution $\mathbb{P}$ of the gene expression data. Dependent on whether x_i is raw or pre-processed, $\mathbb{P}$ can follow a negative binomial or Gaussian distribution. Assuming a mean and variance parametrisation, we get in both cases $\mu = D^\mu_\psi(z')$ and $\sigma^2 = D^{\sigma^2}_\psi(z')$ for the description of the decoded gene expression state. While chemCPA supports both settings, we observed better convergence with a Gaussian likelihood for which the reconstruction loss becomes:

$$\mathcal{L}_{\text{rec}}(\theta, \psi) = N(x_i \,|\, \mu_i, \sigma_i) = \frac{1}{2}\left[\ln\left(D^{\sigma^2}_\psi(z'_i)\right) + \frac{\left(D^\mu_\psi(z'_i) - x_i\right)^2}{D^{\sigma^2}_\psi(z'_i)}\right] \quad \text{with} \quad z' = E_\theta(x) + z_{\text{attribute}} \,,$$

Next, we provide intuition about how we can meaningfully interpret latent space arithmetics and how we encode drug and cell-line attributes.

3.2 Attribute embedding

Due to their different nature, we encode the drug attributes and cell-line information separately. For the cell-lines, we use the same approach as Lotfollahi et al. (2021), where a d-dim latent representation z_c is optimised for each cell-line c.

Perturbation Network For the drugs, we train an embedding network P_φ that maps molecular representations—such as its graph or SMILES representation—and the used dosage to its latent perturbation state. This perturbation network P_φ consists of the molecule and perturbation encoders, G and M, as well as the dosage scaler S, see Figure 1 (2).

The molecule encoder $G : \mathcal{G} \to \mathbb{R}^m$ encodes the molecule representation $g_i \in \mathcal{G}$ as a fixed size embedding $h_{d_i} \in \mathbb{R}^m$. In a subsequent step, the perturbation encoder $M : \mathbb{R}^m \to \mathbb{R}^l$ takes the molecular embedding h_{d_i} as input and generates the drug perturbation $z_{d_i} \in \mathbb{R}^l$ that is used in chemCPA's latent space.

The dosage scaler $S : \mathbb{R}^{m+1} \to \mathbb{R}$ also uses h_{d_i} and maps it together with the dosage s_i to the scaled dosage value $\hat{s}_i$. We chose S to map back to a scalar value $\hat{s}$ as this allows us to compute drug-response curves in an easy fashion. In addition, this way of encoding matches the idea that z_{d_i} encodes the drug's general effect, which is dosage independent. Put together, we end up with

$$\hat{s}_i \times z_{d_i} = P_\varphi(g_i, s_i) = S(h_{d_i}, s_i) \times M(h_{d_i}) \text{ with } h_{d_i} = G(g_i) \quad .$$

The molecule encoder G can be any encoding network that maps molecular representations to fixed-size embeddings. Due to the limited number of drugs available in scRNA-seq HTSs, we propose to rely on a pretrained encoding model and freeze G during training. We tested multiple different options for G and include a detailed benchmark in the appendix A.1. We found that in our setting RDKit features performed really well and report all following results for chemCPA with RDKit as the molecule encoder G.

Additive latent space We assume an additive structure of the perturbation response in the latent space. This choice of linearity makes the model interpretable for users such as biologists since it permits to analyze and ablate components individually, e.g., allowing interpolation or extrapolation of the dose values. Another advantage of the additive structure is its permutation invariance and that it allows for adding new covariates, e.g., during fine-tuning. While remaining interpretable, chemCPA is able to model complex relationships through the non-linear decoder.

By design, we can choose a new set of attributes for the drug and cell line and compute the new latent state as $z_i' = z_i + z_{\text{attribute}}$ at test time. Due to the perturbation network, chemCPA makes it possible to predict drug perturbations for molecules that have not been experimentally observed ($d \notin \mathcal{D}$). In contrast, CPA can only make predictions for molecules that were present during training ($d \in \mathcal{D}$). In both cases, the latent representation is computed as:

$$z_i' = z_i + z_{\text{attribute}} = z_i + z_{c_i} + \hat{s}_i z_{d_i} \ .$$

We next describe how we "strip z_i from its attribute information" to obtain a basal state representation.

3.3 Adversarial classifiers for invariant basal states

To generate invariant basal states and produce disentangled representations z_i, z_{d_i}, and z_{c_i}, we use adversarial classifiers A_ϕ^{drug} and A_ϕ^{cov}. Both adversary networks $A_\phi^j : \mathbb{R}^l \to \mathbb{R}^{N_j}$ take the latent basal state z_i as input and aim to predict the drug that has been applied to example i as well as its cell-line c_i. While these classifiers are trained to improve classification performance, we also add the classification loss *with a reversed sign* to the training objective for the encoder E_θ. Hence, the encoder attempts to produce a latent representation z_i *which contains no information about the attributes*. Note that this explicit separation of basal, drug, and covariate information, which we call disentanglement, is an approximation to make the problem tractable. At the same time, such separation is useful for attributing perturbation effects to specific sources, e.g., drug or cell line, which is relevant for biological applications and downstream analyses.

We use the cross-entropy loss for both classifiers

$$\mathcal{L}_{\text{class}}^{\text{drugs}} = \text{CE}\big(A_\phi^{\text{drug}}(z_i)\big), d_i) \quad \text{and} \quad \mathcal{L}_{\text{class}}^{\text{cov}} = \text{CE}\big(A_\phi^{\text{cov}}(z_i), c_i\big) \ .$$

Following the CPA implementation from Lotfollahi et al. (2021), we add a zero-centered gradient penalty to the loss function of the adversarial classifiers, to minimise

$$\mathcal{L}_{\text{pen}}^j = \frac{1}{k} \sum_k \left\| \partial_{z_i} A_\phi^j(z_i)_k \right\|_2^2 .$$

This gradient penalty was shown to make the discriminator more robust to noise and enable local convergence, when applied to generative adversarial networks (Mescheder et al., 2018). During training, we alternate update steps between the following competing objectives

$$\mathcal{L}_{\text{AE}}(\theta, \psi, \varphi | \phi) = \mathcal{L}_{\text{rec}}(\theta, \psi, \varphi) - \lambda_{\text{dis}} \sum_j \mathcal{L}_{\text{class}}^j(\theta \,|\, \phi) \quad \text{and}$$

$$\mathcal{L}_{\text{Adv}}(\phi \,|\, \theta) = \sum_j \mathcal{L}_{\text{class}}^j(\phi \,|\, \theta) + \lambda_{\text{pen}} \mathcal{L}_{\text{pen}}^j(\phi) \ ,$$

where λ_{dis} balances the importance of good reconstruction against the encoder E_θ's constraint to generate disentangled basal states z_i. The gradient penalty is weigthed with λ_{pen}.

4 Datasets and transfer learning

We use the sci-Plex3 (Srivatsan et al., 2020) and the L1000 (Subramanian et al., 2017) datasets for the main evaluation on single-cell data and pretraining on bulk experiments, respectively.

Datasets The L1000 data contains about 1.3 million bulk RNA observations for 978 different genes. It includes measurements for almost 20k different drugs, some of which are FDA-approved, while others are synthetic compounds with no proven effect on any disease. Compared to scRNA-seq data, the L1000 data allows to explore a more diverse space of molecules which makes it ideal for pretraining.

The sci-Plex3 data is similar in size and contains measurements for 649,340 cells across 7561 drug-sensitive genes. On three human cancer cell lines—A549, MCF7, and K562—single-compound perturbations for 188 drugs at four different dosages— 10 nM, 100 nM, 1 μM, and 10 μM— are examined. Note that all cell lines and about 150 compounds overlap with the L1000 data. In addition, Srivatsan et al. (2020) assigned to all compounds one of 19 different modes of action (MoA), also called pathways. In contrast to the mechanism of action, which is related to the biochemical interaction between a molecule and a cell, the MoA describes the anatomical change that results from the exposure of cells to a drug-like molecule.

Transfer learning As we train chemCPA with a Gaussian likelihood loss, the dataset was first normalised and then $\log(x+1)$-transformed. Depending on the experiment, we further reduced the number of genes included in the single-cell data. In Section 5.1, we first subsetted both datasets to the same 977 genes which were identified via ensemble gene annotations. For the final experiment in Section 5.2, the considered gene set is increased as we hypothesize that more than the 977 L1000 genes are required to capture the variability within the single-cell data. To assess whether pretraining on L1000 is still beneficial in this scenario, we included 1023 highly variable genes (HVGs) from the sci-Plex3 data. That is, we consider 2000 genes in total.

For the extended gene set, chemCPA's input and output dimensions have to be adjusted to match the total number of 2000 genes. This is realized by adding two non-linear layers $h_{\text{enc}} : \mathbb{R}^{n_{\text{finetune}}} \to \mathbb{R}^{n_{\text{pretrain}}}$ and $h_{\text{dec}} : \mathbb{R}^{2n_{\text{pretrain}}} \to \mathbb{R}^{2n_{\text{finetune}}}$ to the autoencoder. The encoder becomes $\hat{E}_\theta = E_\theta(h_{\text{enc}}(x))$ and the decoder becomes $\hat{D}_\psi = h_{\text{dec}}(D_\psi(z'))$. In our example, $n_{\text{pretrain}} = 977, n_{\text{finetune}} = 2000$. We train all layers during finetuning, including the newly added ones. This architecture surgery differs from the procedure introduced in Lotfollahi et al. (2022), where individual neurons are added (instead of whole layers) and the transfer is perform on dataset labels (instead of gene sets).

5 Experiments

Our evaluation strategy tests chemCPA's ability to produce counterfactual predictions. To this end, it is important to measure both the predictive performance of a trained model as well as the degree to which the latent space components are disentangled.

Counterfactual predictions To perform a counterfactual prediction, chemCPA first encodes an unperturbed control observations, a cell treated with dimethyl sulfoxide. The resulting basal state is then combined with the encoding of the desired drug and cell line, and chemCPA decodes the result. As we are free to choose any drug encoding, we refer to this process as counterfactual prediction.

Evaluation strategy Throughout our experiments, we use the coefficient of determination r^2 as the main performance metric. This score is computed between the actual measurements and the counterfactual predictions on all genes and the 50 most differentially expressed genes (DEGs). It is necessary to consider all genes to evaluate the background and general decoder performance. However, the resulting r^2-scores can get inflated since most genes stay similar to their controls under perturbation. In contrast, the DEGs capture the differential signal which reflects a drug's effect. To further stress the importance to report both scores, note that the DEGs are unknown for unseen drugs as they depend on the drug and cell type. Hence, the combination is essential to gauge the accuracy of the model's predictions.

Table 1: Comparison of multiple models on their performance on generalisation to unseen drug-covariate combinations for dosage values of 1 μM and 10 μM.

Dose	Model	$\mathbb{E}[r^2]$ all	$\mathbb{E}[r^2]$ DEGs	Median r^2 all	Median r^2 DEGs
1 μM	Baseline	0.69	0.51	0.82	0.62
	scGen	0.73	0.59	0.77	0.68
	CPA	0.72	0.54	**0.86**	0.67
	chemCPA	0.74	0.60	**0.86**	0.66
	chemCPA pretrained	**0.77**	**0.68**	0.85	**0.76**
10 μM	Baseline	0.50	0.29	0.48	0.12
	scGen	0.62	0.47	0.66	0.49
	CPA	0.54	0.34	0.52	0.26
	chemCPA	0.71	0.58	0.77	0.64
	chemCPA pretrained	**0.76**	**0.68**	**0.82**	**0.79**

In order to classify the degree of disentanglement during evaluation, we train separate MLPs with four layers over 400 epochs and compute the prediction accuracy for drugs and covariates given the basal state. We consider the resulting accuracies as our disentanglement scores. An optimally disentangled model achieves scores that match the ratios of the most abundant drug and cell line, respectively. Since no model achieves perfect scores, we subset to models that are sufficiently disentangled. Throughout our experiments on the sci-Plex3 data, we set the thresholds for perturbation and cell line disentanglement to $< 10\%$ and $< 70\%$, respectively, while values of 3% and 51% are optimal. Note that poor disentanglement will automatically lead to low scores due to computing the test scores on the counterfactual predictions.

5.1 Comparing chemCPA against existing methods on unseen drug-covariate combinations

Before evaluating how well chemCPA can generalise to unseen drugs, we have to establish its competitive performance in a less ambitious setting. For this, we consider the scenario of generalisation to unobserved combinations of drugs and cell lines on the sci-Plex3 data and compare chemCPA against scGen (Lotfollahi et al., 2019) and chemCPA (Lotfollahi et al., 2021).

As scGen cannot distinguish between different dosage values, we perform two separate experiments for the second and highest dose values, 1 μM and 10 μM, respectively. Moreover, as both CPA and scGen require each individual component (drug d, cell line c) to be part of the training data $\mathcal{D}$, we create three distinct splits with only two of the three different drug set and cell line combinations being present during training, the third one being left for testing.

We choose to test nine different compounds: Dacinostat, Givinostat, Belinostat, Hesperadin, Quisinostat, Alvespimycin, Tanespimycin, TAK-901, and Flavopiridol. These drugs mostly belong to three MoA—epigenetic regulation, tyrosine kinase signalling, and cell cycle regulation—and were reported among the most effective drugs in the original publication (Srivatsan et al., 2020).

As discussed in the previous paragraph, we report mean and median r^2 values, which we averaged over the three splits and all drugs, for the sets of all genes and differentially expressed genes (DEGs). We consider a model that discards all perturbation information as our baseline. As a consequence, one can understand the improvement over the baseline as a result of the additional drug encoding. Moreover, we use the L1000 data for the pretraining of chemCPA and subset to the same 977 genes for the fine-tuning on sciPlex-3. This way, we are able to evaluate chemCPA in its pretrained and non-pretrained version.

We report the results of this experiment in Table 1. ChemCPA outperforms both scGen and CPA, demonstrating that the perturbation network together with pretraining leads to SOTA performance. Note also that the base version of chemCPA performs better than both CPA and scGen, indicating that the additional regularisation that comes from the perturbation networks P_φ has beneficial effects on single-cell perturbation modelling.

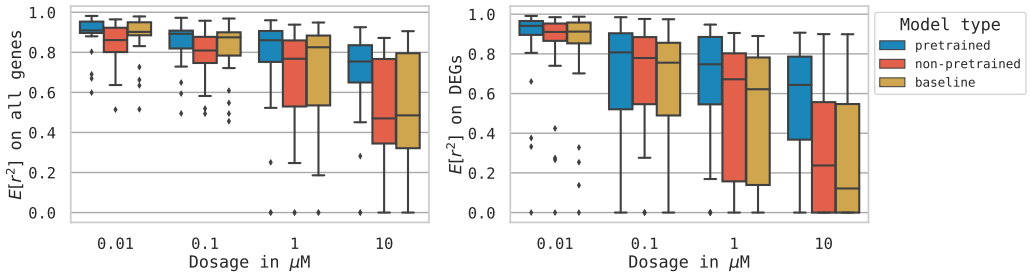

Figure 2: Performance of chemCPA on both the complete gene set (977 genes) and the compound specific DEGs (50 genes). In both cases, the pretrained model shows the best performance. At $10\,\mu$M on the DEGs, more than 50% of the predictions have an r^2 score > 0.6 while the baseline's median is below 0.2.

Table 2: Performance of chemCPA on the shared gene set. Since drug effects are stronger for high dosages, we present scores for a dosage value of $10\,\mu$M.

Model	$\mathbb{E}[r^2]$ all	$\mathbb{E}[r^2]$ DEGs	Median r^2 all	Median r^2 DEGs
Baseline	0.50	0.29	0.49	0.12
chemCPA	0.51	0.32	0.47	0.24
chemCPA pretrained	**0.68**	**0.54**	**0.75**	**0.64**

To make a fair comparison, we optimised all CPA and chemCPA models identically and swept over the same set of hyperparameters (random, 10 samples). scGen was optimised with default parameters and an adjusted KL annealing scheme to match the set number of epochs. Note that both CPA and chemCPA can take control cells x_i from all cell lines as input ($\{$A549, K562, MCF7$\} \rightarrow \{$A549, K562, MCF7$\}$), while the cell line input for scGen has to match the test set (A549 $\rightarrow$ A549, K562 $\rightarrow$ K562, MCF7 $\rightarrow$ MCF7).

5.2 Using chemCPA to predict single-cell responses for unseen drugs

For the application of chemCPA to predict perturbation responses for unseen compounds, we use the same nine drugs from sci-Plex3 data as in Section 5.1. In addition to the shared gene set, we also consider an extended gene set, cf. Section 4. We include HVGs to account for the technological difference between bulk and single-cell and to capture the variance of single-cell data. To this end, the 977 genes present in both datasets are extended with 1023 HVGs of the sci-Plex3 data. Note that through this larger genes set, the 50 DEGs become a subset of the HVGs which makes it considerably more difficult for pretrained models to leverage learned bulk expressions directly.

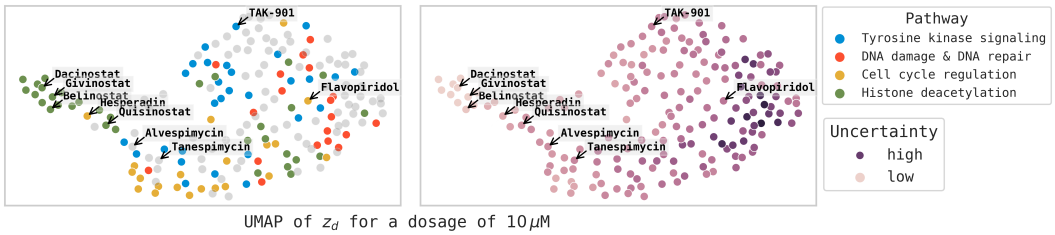

Figure 3: Illustration of the the scaled perturbation embedding $s \times z_d$ for $10\,\mu$M. The left part illustrates how the perturbation embeddings z_d are clustered according to some of the pathways. Most notably, the histone deacetylation drugs show a clear separation. Further context is provided by the uncertainty score on the right, showing regions of high and low confidence for the drug embeddings z_d. The nine test compounds are labeled.

Table 3: We show the performance of chemCPA on the extended gene set. Since drug effects are stronger for high dosages, we present scores for a dosage value of 10 μM.

Model	$\mathbb{E}[r^2]$ all	$\mathbb{E}[r^2]$ DEGs	Median r^2 all	Median r^2 DEGs
Baseline	0.37	0.19	0.16	0.00
chemCPA	0.46	0.22	0.35	0.00
chemCPA pretrained	**0.69**	**0.47**	**0.79**	**0.62**

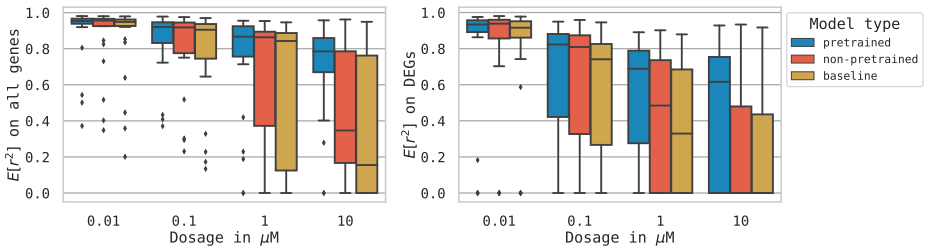

Figure 4: Performance of chemCPA model on the extended gene set, see also Figure 2. The pretrained model shows the best performance with the non-pretrained model failing to beat the baseline at lower dosages. At 10 μM on the DEGs, the pretrained model's median much higher that the baseline.

Shared gene set Table 2 shows the test performance of chemCPA, averaged over all drugs and the three cell lines, for the same gene set as used in Section 5.1. The pretrained chemCPA model consistently outperform the baseline and its base version.

The high baseline scores in Figure 2 shows that the drugs have almost no effect at low dosages. At high dosages, however, we see how chemCPA's predictions improve over the baseline. Looking at the prediction for all genes, the pretrained model has a significant advantage over its non-pretrained version. As expected, the performance is lower for the DEGs. Nevertheless, also in this scenario, chemCPA, especially the pretrained version, can explain gene expression values that must result from the drugs' influence.

In Figure 3, latent perturbations z_d are visualised. Note that the difference between the lowest and highest doseage values results only from the non-linear dosage scaler S.

Extended gene sets The extension to the larger gene set introduces a more difficult task for chemCPA. In Table 3, we show the same analysis as for the shared gene set. Again, the advantage of the pretraied chemCPA model translates to this scenario, while the base version is only slightly better than the baseline. A more comprehensive view for all dosages is shown in Figure 4, see also A.5 for results with different molecule encoders G.

This is a promising result, as it suggests that the transfer from abundant bulk RNA perturbation screens can be leveraged even in scenarios where the gene sets do not match. Crucially, this enables users to benefit from the proposed transfer learning and chemCPA's modelling capacity while simultaneously accounting for the special requirements of scRNA-seq data. In Figure 5 we show an explicit example for chemCPA's performance for two histone deacetylation drugs, see also Figure 13 in the appendix for more details.

Yet, for real applications it is essential to make limitations concerning the data and method transparent. To this end, we propose an uncertainty measure that addresses some of the limitations related to the generalisation to unseen compounds.

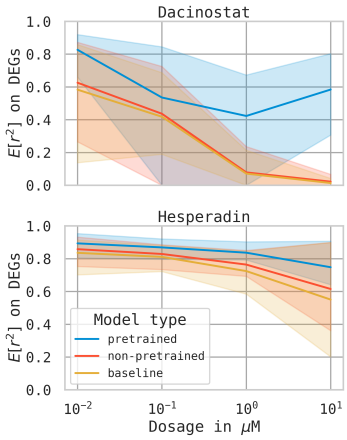

Figure 5: Perturbation prediction for Dacinostat and Hesperadin for chemCPA across all three cell-lines for the shared gene set.

Table 4: Uncertainty score for all nine unseen drugs. The last column shows the improvement Δr^2 of chemCPA over the baseline. The number of considered neighbours was nine for all drugs.

Drug	Uncertainty score u	Δr^2 on DEGs
Dacinostat	0.570	0.00
Givinostat	0.660	0.00
Belinostat	0.623	0.12
Hesperadin	0.197	0.40
Quisinostat	0.052	0.65
Alvespimycin	0.058	0.74
Tanespimycin	0.092	0.75
TAK-901	0.049	0.88
Flavopiridol	0.011	0.99

5.3 Measure uncertainty on the drug embedding

Generalisation can only be achieved within the limits of the dataset a model is trained on. For the sci-Plex3 data, less than 20% of the drug-dose combinations deviate from the controls' phenotypes by more than 35% in its r^2 scores. In additon, we know from the original sci-Plex3 publication that only drugs from a few pathways—tyrosine kinase signaling, DNA damage and repair, cell cycle regulation, and epigenetic regulation—show a clear effect. We assume that this technological noise is the reason why the non-pretrained chemCPA version struggles to outperform the baseline on the extended gene set, whereas the pretrained model is more robust. These data challenges are also reflected in the left part of Figure 3 as we would expect chemCPA's perturbation latent space to cluster according to the drugs' MoA, similar to the cluster of histone deacetylation drugs.

We found that an imperfect clustering often correlates with high baseline scores and, as a result of that, chemCPA not being able to identify distinct perturbations, see Figure 15 in the appendix. To make the generalisation ability more transparent, we employ a measure of uncertainty. A good indicator for chemCPA's ability to generalise is the MoA prediction from the KNN-graph of the perturbation embedding space. We further combine this measure with the average distance to neighbouring drugs as we recognise that larger distances indicate a distinct perturbation:

$$u_i = \sum_{j \in \mathcal{N}_i} \frac{1}{\log\big(d(i,j)\big)} \times H(X) \quad ,$$

where d is the Euclidean distance, H is the Shannon entropy, and X the normalised pathway prediction deduced from the neighbours $\mathcal{N}_i$ of drug i. This uncertainty measure combines two things: First, chemCPA's confidence on the drug's MoA, measured by H, and second, whether chemCPA expects the drug to have a distinct perturbation effect on the cell, measured by the inverse distance.

We report an analysis of the uncertainty for the chemCPA model in Figure 3 and Table 4. A plot that shows chemCPA's performance and uncertainty for all compounds is part of the appendix A.6. The results show that the uncertainty score u for unseen drugs correlates well with the accurate prediction of perturbed cells. This illustrates how chemCPA's compositional latent space can be leveraged for additional insights in order to evaluate its generalisation ability.

6 Conclusion

In this paper, we introduced chemCPA, a model for predicting cellular gene expression responses for unseen drug perturbations by encoding the drugs' molecular structures. We showed how chemCPA outperforms CPA and scGen on shared tasks, while generalising over existing methods by being applicable to the novel task of generalising to unseen drugs. Applied to single-cell data, we demonstrated how pretraining on bulk HTSs improves chemCPA's generalisation performance. This applies even when the gene set of the single-cell dataset differs from the genes of the pretraining bulk RNA HTS dataset. We further provided an uncertainty measurement that correlates well with the chemCPA's generalisation ability to unseen drugs. Taken all results together, we are confident that chemCPA will benefit from higher-quality scRNA-seq HTSs in the future and can become a powerful aid in the drug screening and drug discovery process.

References

Matthew Amodio, Dennis Shung, Daniel B Burkhardt, Patrick Wong, Michael Simonov, et al. Generating hard-to-obtain information from easy-to-obtain information: applications in drug discovery and clinical inference. *Patterns*, 2021.

Atray Dixit, Oren Parnas, Biyu Li, Jenny Chen, Charles P Fulco, et al. Perturb-seq: dissecting molecular circuits with scalable single-cell rna profiling of pooled genetic screens. *cell*, 2016.

Fabian Fröhlich, Thomas Kessler, Daniel Weindl, Alexey Shadrin, Leonard Schmiester, et al. Efficient parameter estimation enables the prediction of drug response using a mechanistic pan-cancer pathway model. *Cell systems*, 2018.

Adam Gayoso, Romain Lopez, Galen Xing, Pierre Boyeau, Valeh Valiollah Pour Amiri, et al. A python library for probabilistic analysis of single-cell omics data. *Nature Biotechnology*, 2022.

Jase Gehring, Jong Hwee Park, Sisi Chen, Matthew Thomson, and Lior Pachter. Highly multiplexed single-cell rna-seq for defining cell population and transcriptional spaces. *BioRxiv*, 2018.

Xiaoping Han, Ziming Zhou, Lijiang Fei, Huiyu Sun, Renying Wang, et al. Construction of a human cell landscape at single-cell level. *Nature*, 2020.

Yuge Ji, Mohammad Lotfollahi, F Alexander Wolf, and Fabian J Theis. Machine learning for perturbational single-cell omics. *Cell Systems*, 2021.

Kenji Kamimoto, Christy M Hoffmann, and Samantha A Morris. Celloracle: Dissecting cell identity via network inference and in silico gene perturbation. *bioRxiv*, 2020.

Guillaume Lample, Neil Zeghidour, Nicolas Usunier, Antoine Bordes, Ludovic Denoyer, et al. Fader networks: Manipulating images by sliding attributes. *Advances in neural information processing systems*, 2017.

Romain Lopez, Adam Gayoso, and Nir Yosef. Enhancing scientific discoveries in molecular biology with deep generative models. *Molecular Systems Biology*, 2020.

Mohammad Lotfollahi, F Alexander Wolf, and Fabian J Theis. scgen predicts single-cell perturbation responses. *Nature methods*, 2019.

Mohammad Lotfollahi, Anna Klimovskaia Susmelj, Carlo De Donno, Yuge Ji, Ignacio L Ibarra, et al. Learning interpretable cellular responses to complex perturbations in high-throughput screens. *bioRxiv*, 2021.

Mohammad Lotfollahi, Mohsen Naghipourfar, Malte D Luecken, Matin Khajavi, Maren Büttner, et al. Mapping single-cell data to reference atlases by transfer learning. *Nature Biotechnology*, 2022.

Christopher S McGinnis, David M Patterson, Juliane Winkler, Daniel N Conrad, Marco Y Hein, et al. Multi-seq: sample multiplexing for single-cell rna sequencing using lipid-tagged indices. *Nature methods*, 2019.

Lars Mescheder, Andreas Geiger, and Sebastian Nowozin. Which training methods for gans do actually converge? In *International conference on machine learning*, pp. 3481–3490. PMLR, 2018.

Thomas M Norman, Max A Horlbeck, Joseph M Replogle, Alex Y Ge, Albert Xu, et al. Exploring genetic interaction manifolds constructed from rich single-cell phenotypes. *Science*, 2019.

Thai-Hoang Pham, Yue Qiu, Jucheng Zeng, Lei Xie, and Ping Zhang. A deep learning framework for high-throughput mechanism-driven phenotype compound screening and its application to covid-19 drug repurposing. *Nature machine intelligence*, 2021.

Ladislav Rampášek, Daniel Hidru, Petr Smirnov, Benjamin Haibe-Kains, and Anna Goldenberg. Dr. vae: improving drug response prediction via modeling of drug perturbation effects. *Bioinformatics*, 2019.

Sergei Rybakov, Mohammad Lotfollahi, Fabian J Theis, and F Alexander Wolf. Learning interpretable latent autoencoder representations with annotations of feature sets. In *Machine Learning in Computational Biology (MLCB) meeting*. Cold Spring Harbor Laboratory, 2020.

Lucas Seninge, Ioannis Anastopoulos, Hongxu Ding, and Joshua Stuart. Vega is an interpretable generative model for inferring biological network activity in single-cell transcriptomics. *Nature communications*, 2021.

Lisa Sikkema, Daniel C Strobl, Luke Zappia, Elo Madissoon, Nikolay S Markov, et al. An integrated cell atlas of the human lung in health and disease. *bioRxiv*, 2022. doi: 10.1101/2022.03.10.483747.

Sanjay R Srivatsan, José L McFaline-Figueroa, Vijay Ramani, Lauren Saunders, Junyue Cao, et al. Massively multiplex chemical transcriptomics at single-cell resolution. *Science*, 2020.

Marlon Stoeckius, Shiwei Zheng, Brian Houck-Loomis, Stephanie Hao, Bertrand Z Yeung, et al. Cell hashing with barcoded antibodies enables multiplexing and doublet detection for single cell genomics. *Genome biology*, 2018.

Aravind Subramanian, Rajiv Narayan, Steven M Corsello, David D Peck, Ted E Natoli, et al. A next generation connectivity map: L1000 platform and the first 1,000,000 profiles. *Cell*, 2017.

Ramzan Umarov, Yu Li, and Erik Arner. Deepcellstate: An autoencoder-based framework for predicting cell type specific transcriptional states induced by drug treatment. *PLoS Computational Biology*, 2021.

Ido Yofe, Rony Dahan, and Ido Amit. Single-cell genomic approaches for developing the next generation of immunotherapies. *Nature medicine*, 2020.

Bo Yuan, Ciyue Shen, Augustin Luna, Anil Korkut, Debora S Marks, et al. Cellbox: interpretable machine learning for perturbation biology with application to the design of cancer combination therapy. *Cell systems*, 2021.

Jie Zhu, Jingxiang Wang, Xin Wang, Mingjing Gao, Bingbing Guo, et al. Prediction of drug efficacy from transcriptional profiles with deep learning. *Nature biotechnology*, 2021.

Checklist

The checklist follows the references. Please read the checklist guidelines carefully for information on how to answer these questions. For each question, change the default **[TODO]** to [Yes] , [No] , or [N/A] . You are strongly encouraged to include a **justification to your answer**, either by referencing the appropriate section of your paper or providing a brief inline description. For example:

- Did you include the license to the code and datasets? [Yes] See Section **??**.
- Did you include the license to the code and datasets? [No] The code and the data are proprietary.
- Did you include the license to the code and datasets? [N/A]

Please do not modify the questions and only use the provided macros for your answers. Note that the Checklist section does not count towards the page limit. In your paper, please delete this instructions block and only keep the Checklist section heading above along with the questions/answers below.

1. For all authors...
   (a) Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope? [Yes] See section claim 1 is supported by 3, claim 2 by A.1, claim 3 by 5.2 and 5.2.
   (b) Did you describe the limitations of your work? [Yes] See section 5.2 and 5.3
   (c) Did you discuss any potential negative societal impacts of your work? [No] We do not foresee any potential harm by this work.
   (d) Have you read the ethics review guidelines and ensured that your paper conforms to them? [Yes]

2. If you are including theoretical results...
   (a) Did you state the full set of assumptions of all theoretical results? [N/A] Not applicable
   (b) Did you include complete proofs of all theoretical results? [N/A] Not applicable
3. If you ran experiments...
   (a) Did you include the code, data, and instructions needed to reproduce the main experimental results (either in the supplemental material or as a URL)? [No] Code link will be included after double-blind review.
   (b) Did you specify all the training details (e.g., data splits, hyperparameters, how they were chosen)? [Yes]
   (c) Did you report error bars (e.g., with respect to the random seed after running experiments multiple times)? [Yes] Top 5 models that met disentanglement criteria in Table **??**, for the other experiments the best performing model according to the validation split was taken.
   (d) Did you include the total amount of compute and the type of resources used (e.g., type of GPUs, internal cluster, or cloud provider)? [No] The experiment files will be made public with the code release
4. If you are using existing assets (e.g., code, data, models) or curating/releasing new assets...
   (a) If your work uses existing assets, did you cite the creators? [Yes]
   (b) Did you mention the license of the assets? [N/A]
   (c) Did you include any new assets either in the supplemental material or as a URL? [Yes] Code link will be included after double-blind review.
   (d) Did you discuss whether and how consent was obtained from people whose data you're using/curating? [N/A] Not applicable
   (e) Did you discuss whether the data you are using/curating contains personally identifiable information or offensive content? [No] All data comes from cell lines.
5. If you used crowdsourcing or conducted research with human subjects...
   (a) Did you include the full text of instructions given to participants and screenshots, if applicable? [N/A] Not applicable
   (b) Did you describe any potential participant risks, with links to Institutional Review Board (IRB) approvals, if applicable? [N/A] Not applicable
   (c) Did you include the estimated hourly wage paid to participants and the total amount spent on participant compensation? [N/A] Not applicable

A Appendix

A.1 Benchmarking drug molecule encoders

Enabled by the flexibility of the molecule encoder G, we investigated what impact the architecture choice has on the performance of chemCPA. For this, we compared multiple pretrained graph-based models whose weights were frozen during the training. Next to predefined RDKit fingerprints (which are non-differentiable, and hence not trainable), we included a GCN, MPNN, weave model, GROVER model, and a JT-VAE.

Table 5 summarises the results of this experiment on the L1000 dataset. The weave model performs much worse than the others, achieving an r^2-score of only 65 ± 8 on DEGs. While the GCN disentangles well, it is outperformed by the JT-VAE and GROVER models. All experiments in the main text were ran across all three best performing models (GROVER, JT-VAE and RDKit), however due to space constraints we only report RDKit results in the main text.

Table 6 shows the test performance of chemCPA for the nine unseen drugs across the three cell lines in the transfer learning scenario with identical genes. This is the same experiment as Table 2, but evaluated across more embedding models. The fine-tuned chemCPA models for GROVER and RDKit consistently outperform the baseline and their non-pretrained version with RDKit achieving the highest median score on DEGs. Interestingly, the fine-tuned JT-VAE model is better than the baseline and other non-pretained chemCPA models but worse than its own non-pretrained version.

Table 5: Summary of chemCPA on the L1000 dataset for different molecule encoders G. All models were trained on the same random split. Reported are the overall disentanglement scores (drug and cell line) and the r^2-scores on the test set.

Model G	Drug	Cell line	Mean r^2 all	Mean r^2 DEGs
GCN	$\mathbf{0.08 \pm 0.03}$	$\mathbf{0.17 \pm 0.01}$	0.92 ± 0.01	0.81 ± 0.05
MPNN	0.10 ± 0.03	0.28 ± 0.07	0.92 ± 0.01	0.82 ± 0.03
GROVER	0.09 ± 0.03	0.19 ± 0.04	0.93 ± 0.01	$\mathbf{0.87 \pm 0.01}$
JT-VAE	$\mathbf{0.08 \pm 0.02}$	0.20 ± 0.04	0.93 ± 0.01	$\mathbf{0.87 \pm 0.01}$
RDKit	0.10 ± 0.04	0.29 ± 0.13	0.93 ± 0.01	0.85 ± 0.03
weave	0.10 ± 0.03	0.29 ± 0.09	0.89 ± 0.02	0.65 ± 0.08

Table 6: Performance of pretrained and non-pretrained chemCPA models across the three versions of the molecule encoder G, for the L1000 to SciPlex3 transfer learning experiment with shared gene sets. Since drug effects are stronger for high dosages, the scores are evaluated at a dosage value of 10 μM.

Model G	Type	Mean r^2 all	Mean r^2 DEGs	Median r^2 all	Median r^2 DEGs
	baseline	0.50	0.29	0.49	0.12
GROVER	non-pretrained	0.52	0.32	0.51	0.18
	pretrained	0.63	0.47	0.70	0.49
JT-VAE	non-pretrained	0.60	0.39	0.68	0.42
	pretrained	0.55	0.35	0.55	0.28
RDKit	non-pretrained	0.51	0.32	0.47	0.24
	pretrained	**0.68**	**0.54**	**0.75**	**0.64**

In Table 7, we show the same experiment as in Table 3. Again, the pretrained chemCPA model with an RDKit molecule encoder G perform best. We believe that this can be attributed to two things. First, the sci-Plex3 data is the first of its kind, and technological noise is still an issue. When evaluated over the whole training set, the baseline achieves r^2-scores higher than 65% for more than 96% of the observations. This sparsity might hinder the more complex perturbation networks P_φ, which are based on GROVER and JT-VAE, from finding good perturbation representations. We suspect that the same reason also explains the bad performance of non-pretrained models as these are more susceptible to noise, whereas the fine-tuned models are more robust. Second, the pretrained embedding h that result from RDKit identifies the histone deacetylation drugs as a distinct cluster, see Figure 10. Since these compounds show the strongest effect in the sci-Plex3 data, the inductive bias from RDKit give an explanation for the favourable generalisation performance.

Table 7: Performance of pretrained and non-pretrained chemCPA models across the three versions of the molecule encoder G on the extended gene set. Since drug effects are stronger for high dosages, the scores are evaluated at a dosage value of 10 μM.

Model G	Type	Mean r^2 all	Mean r^2 DEGs	Median r^2 all	Median r^2 DEGs
	baseline	0.37	0.19	0.16	0.00
GROVER	non-pretrained	0.41	0.22	0.28	0.00
	pretrained	0.59	0.36	0.75	0.45
JT-VAE	non-pretrained	0.40	0.22	0.20	0.00
	pretrained	0.51	0.24	0.51	0.00
RDKit	non-pretrained	0.46	0.22	0.35	0.00
	pretrained	**0.69**	**0.47**	**0.79**	**0.62**

A.2 Attribute embedding

Table 8: Details on pretrained models for the molecule encoder G.

Molecule encoder G	Embedding dim h_{drug}	Pretrained
RDKit	200	–
GROVER	3400	authors
JT-VAE	56	ZINC, L1000, sci-Plex3
GCN	128	PCBA
MPNN	128	PCBA
weave	128	PCBA

A.3 Counterfactual prediction

1. To compute counterfactual predictions, we obtain basal states z_i for all control observations present in the test set. For each combination of drug, dose, and cell line in the test set, we compute the latent attribute state $z_{\text{attribute}}$ and combine it with all z_i. Subsequently, we compute the mean per gene across all predictions and likewise for the real measurements. As a result, we obtain two n dimensional vectors, where n is the number of genes (977 or 2000), for which we compute the r^2 score. Taken together, we get one score per combination.

A.4 Additional information on the L1000 experiment

1. For infos on the RDKit sweep and resulting best run, see Table 9 and Table 10.
2. Architectures for best configuration of the perturbation networks P_φ and adversary classifiers are presented in Table 11.
3. For details on the performance of the best runs, see Table 12.

Table 9: Fixed Parameters for the RDKit sweep in the L1000 dataset.

Parameter	Value
num_epochs	1500
dataset_type	lincs
decoder_activation	linear
model	rdkit

A.5 Additional information on the sci-Plex3 experiments

1. The optimisation was performed similarly to the presented sweeps in Table 10 and Table 11 for the perturbation network and adversary parameters for 10 samples each per category.
2. Boxplot results for RDKit, see Figures 6 and 8, and JT-VAE, see Figures 7 and 9.
3. Paired t-tests were performed for both settings, see Table 14 for the shared gene set and Table 14 for the extended gene set.
4. More examples on the performance with respect to specific drugs are presented in Figure 11, Figure 12, Figure 13, and Figure 14.
5. The Drug embedding that results from RDKit is shown in

A.6 Additional information on the uncertainty score

1. The uncertainty computation for the chemCPA model with an RDKit molecule embedding for the shared gene setting is shown in Figure 15.

Table 10: Random parameters for the RDKit sweep in the L1000 dataset.

Parameter	Type	Values	Best config
samples	fixed	25	NaN
dim	choice	{64, 32}	32
dosers_width	choice	{64, 256, 128, 512}	64
dosers_depth	choice	{1, 2, 3}	1
dosers_lr	loguniform	$[1 \times 10^{-4}, 1 \times 10^{-2}]$	5.61×10^{-4}
dosers_wd	loguniform	$[1 \times 10^{-8}, 1 \times 10^{-5}]$	1.33×10^{-7}
autoencoder_width	choice	{128, 256, 512}	256
autoencoder_depth	choice	{3, 4, 5}	4
autoencoder_lr	loguniform	$[1 \times 10^{-4}, 1 \times 10^{-2}]$	1.12×10^{-3}
autoencoder_wd	loguniform	$[1 \times 10^{-8}, 1 \times 10^{-5}]$	3.75×10^{-7}
adversary_width	choice	{64, 256, 128}	128
adversary_depth	choice	{2, 3, 4}	3
adversary_lr	loguniform	$[5 \times 10^{-5}, 1 \times 10^{-2}]$	8.06×10^{-4}
adversary_wd	loguniform	$[1 \times 10^{-8}, 1 \times 10^{-3}]$	4.0×10^{-6}
adversary_steps	choice	{2, 3}	2
reg_adversary	loguniform	[5, 100]	24.1
penalty_adversary	loguniform	[1, 10]	3.35
batch_size	choice	{32, 64, 128}	128
step_size_lr	choice	{200, 50, 100}	100
embedding_encoder_width	choice	{128, 256, 512}	128
embedding_encoder_depth	choice	{2, 3, 4}	3

Table 11: Presented are the best configurations per molecule encoder from 18 random hyperparamter samples similar to the one presented in Table 10.

Parameter	GROVER	MPNN	RDKit
dosers_width	512	64	64
dosers_depth	2	2	3
dosers_lr	5.61×10^{-4}	1.58×10^{-3}	1.12×10^{-3}
dosers_wd	1.33×10^{-7}	6.25×10^{-7}	3.75×10^{-7}
embedding_encoder_width	512	128	128
embedding_encoder_depth	3	4	4
Parameter	**weave**	**JT-VAE**	**GCN**
dosers_width	512	64	512
dosers_depth	2	2	2
dosers_lr	1.12×10^{-3}	2.05×10^{-4}	2.05×10^{-4}
dosers_wd	2.94×10^{-8}	2.94×10^{-8}	1.33×10^{-6}
embedding_encoder_width	128	256	128
embedding_encoder_depth	3	4	3

Table 12: Performance of the best runs on L1000 for different molecule encoders G

Model G	Drug	Cell line	Mean r^2 all	Mean r^2 DEGs	Mean r^2 DEGs [val]
GCN	0.11	0.16	0.92	0.84	0.83
MPNN	0.07	0.24	0.94	0.87	0.84
GROVER	0.07	0.16	0.94	0.88	0.86
JT-VAE	0.06	0.15	0.94	0.88	0.85
RDKit	0.08	0.15	0.93	0.86	0.85
weave	0.09	0.20	0.91	0.74	0.72

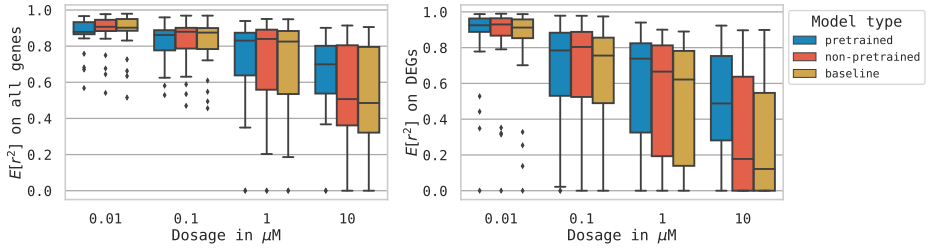

Figure 6: Performance of the pretrained and non-pretrained chemCPA model using GROVER. Comparisons against the baseline are done on both the complete gene set (977 genes) and the compound specific DEGs (50 genes).

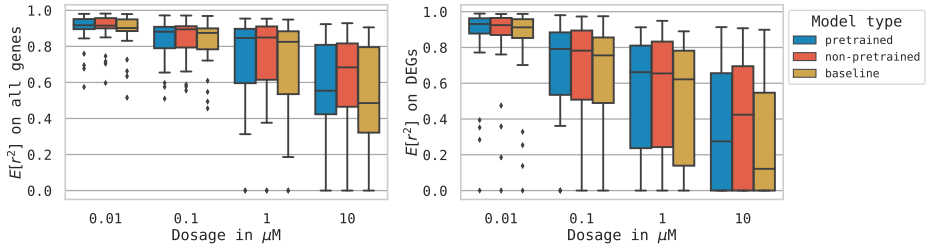

Figure 7: Performance of the pretrained and non-pretrained chemCPA model using JT-VAE. Comparisons against the baseline are done on both the complete gene set (977 genes) and the compound specific DEGs (50 genes).

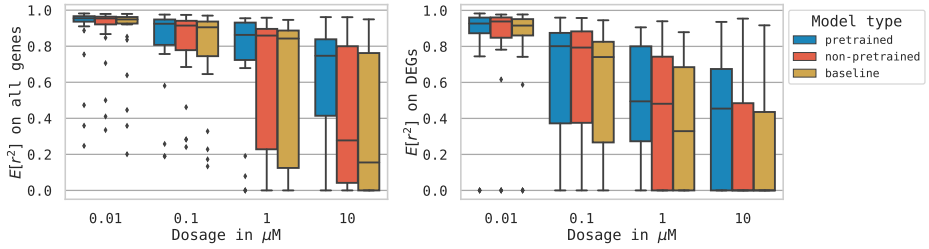

Figure 8: Performance of the pretrained and non-pretrained chemCPA model on the extended gene set using GROVER, see also Figure 6.

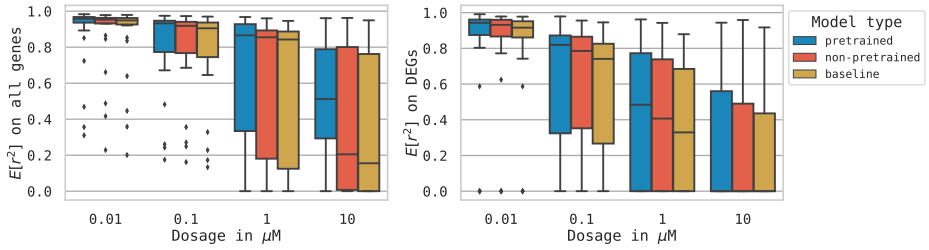

Figure 9: Performance of the pretrained and non-pretrained chemCPA model on the extended gene set using JT-VAE, see also Figure 7.

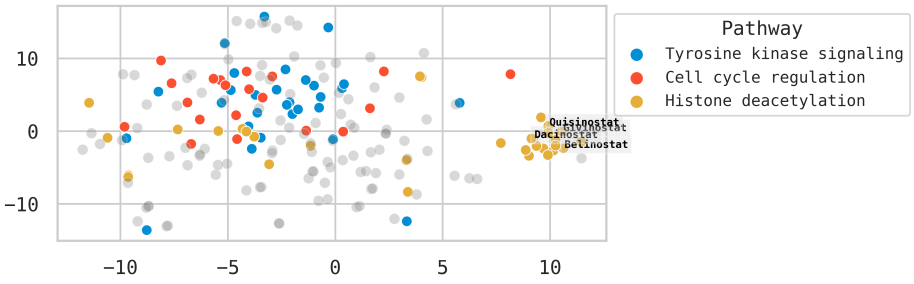

Figure 10: TSNE embedding based on the RDKit features of the 188 drugs.

Table 13: Significance test for a dosage of 10 μM on the shared gene set using the paired t-test.

Model G	Against	Gene set	p-value
rdkit	baseline	all genes	0.0002
rdkit	baseline	DEGs	0.0001
rdkit	non-pretrained	all genes	0.0001
rdkit	non-pretrained	DEGs	0.0003
grover	baseline	all genes	0.0008
grover	baseline	DEGs	0.0002
grover	non-pretrained	all genes	0.0023
grover	non-pretrained	DEGs	0.0022
jtvae	baseline	all genes	0.0002
jtvae	baseline	DEGs	0.0004
jtvae	non-pretrained	all genes	0.0141
jtvae	non-pretrained	DEGs	0.0528

Table 14: Significance test for a dosage of 10 μM on the extended gene set using the paired t-test.

Model G	Against	Gene set	p-value
rdkit	baseline	all genes	0.0001
rdkit	baseline	DEGs	0.0004
rdkit	non-pretrained	all genes	0.0003
rdkit	non-pretrained	DEGs	0.0020
grover	baseline	all genes	0.0009
grover	baseline	DEGs	0.0038
grover	non-pretrained	all genes	0.0029
grover	non-pretrained	DEGs	0.0165
jtvae	baseline	all genes	0.0005
jtvae	baseline	DEGs	0.0024
jtvae	non-pretrained	all genes	0.0026
jtvae	non-pretrained	DEGs	0.0721

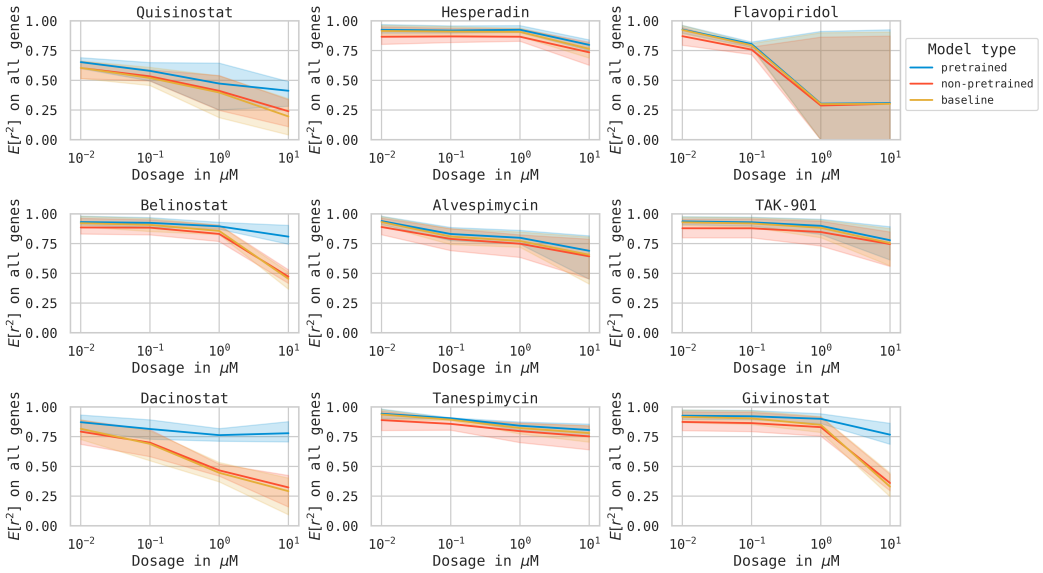

Figure 11: Drug-wise comparison between the baseline, pretrained and non-pretrained models using RDKit for all nine drugs in the test set considering all genes for the shared gene set.

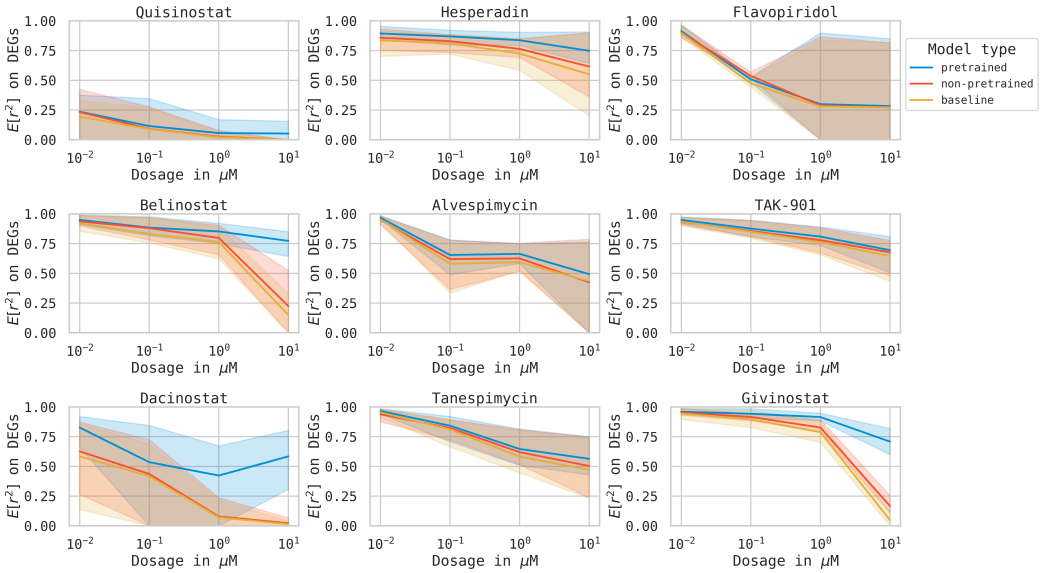

Figure 12: Drug-wise comparison between the baseline, pretrained and non-pretrained models using RDKit for all nine drugs in the test set considering the DEGs for the shared gene set.

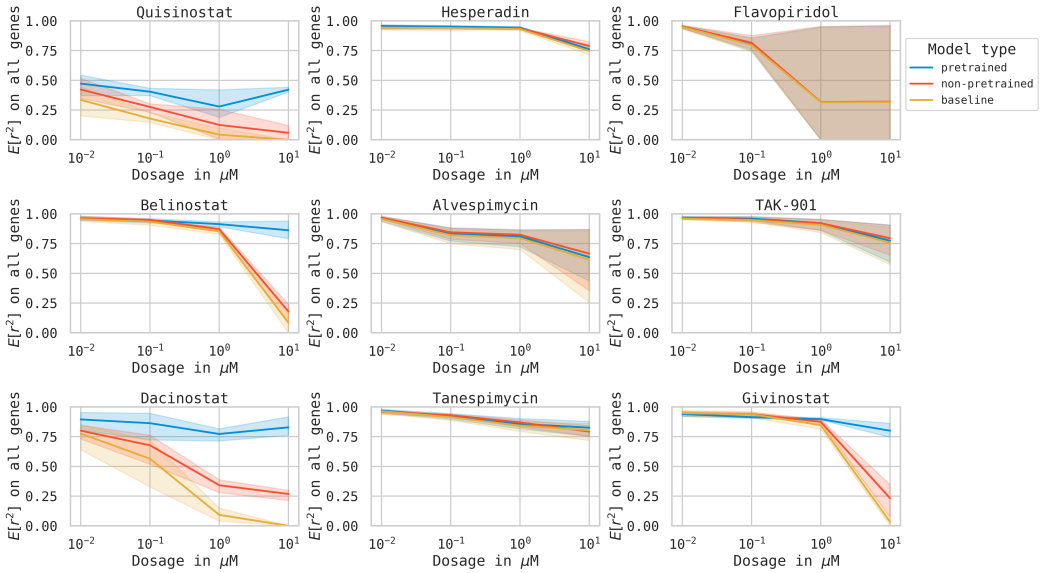

Figure 13: Drug-wise comparison between the baseline, pretrained and non-pretrained models using RDKit for all nine drugs in the test set considering all genes for the extended gene set.

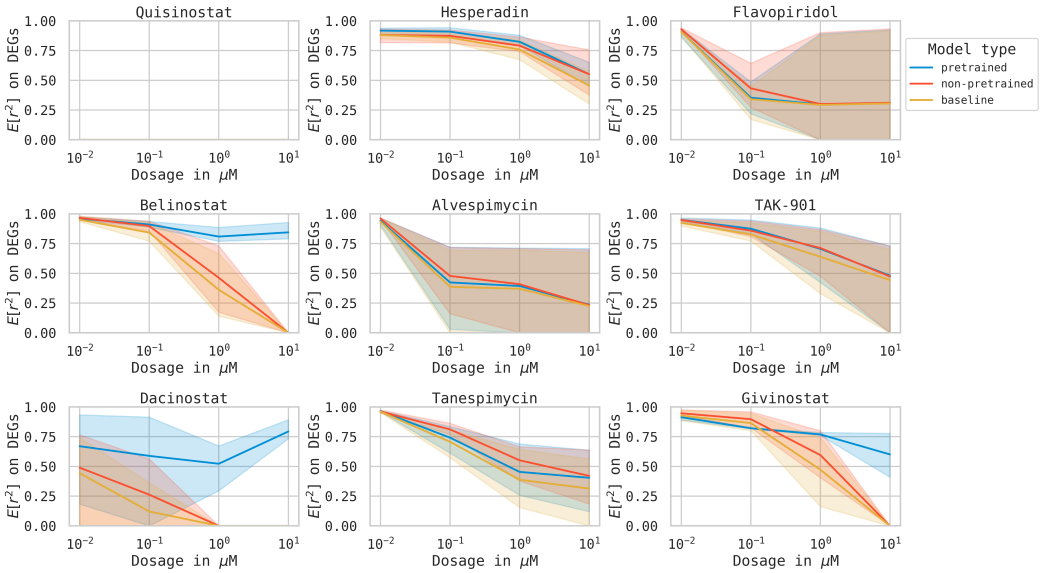

Figure 14: Drug-wise comparison between the baseline, pretrained and non-pretrained models using RDKit for all nine drugs in the test set considering the DEGs for the extended gene set.

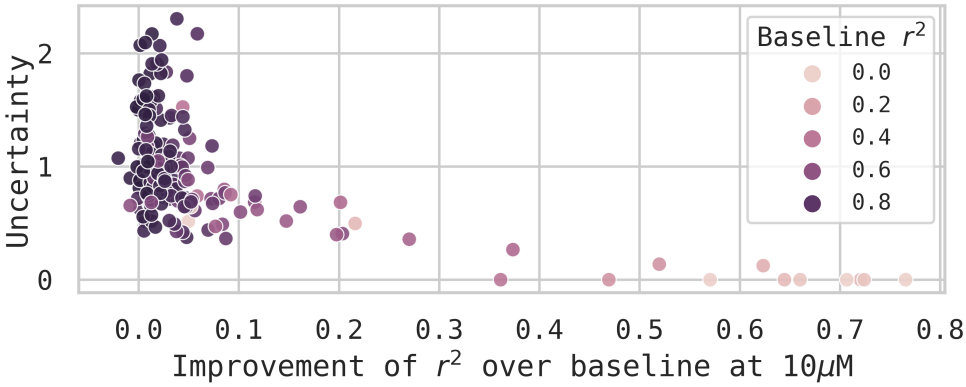

Figure 15: Uncertainty score for chemCPA's prediction on the perturbation embedding in relation to the model's improvement over the baseline score, measured in r^2.