SHARE TO EXTRAPOLATE: LEARNING REUSABLE RESPONSE COMPONENTS FOR SINGLE-CELL PERTURBATION PREDICTION

Anonymous authors
Paper under double-blind review

ABSTRACT

Predicting how genetic perturbations change gene expression is central to understanding gene function and cellular regulation. Yet experimental coverage remains limited, requiring generalization to unseen perturbations. Despite advances in perturbation prediction, matching observed conditional distributions alone leaves responses to unseen perturbations underdetermined. With supervision limited to observed perturbations, models can fit the training distributions equally well yet predict different responses to the same unseen perturbation. Similar patterns of gene-expression change across perturbations suggest that shared response components can provide an inductive bias for extrapolation. We introduce CoShare, a graph-routed factorized Delta Flow Matching framework. CoShare decomposes responses relative to the control mean into mean responses and cell-level residuals, learning from unpaired control and perturbed populations. A component router uses multi-source gene graphs and cell-type context to produce a component coefficient vector. This vector weights shared response vectors in the population mean branch and controls adapters combining shared feature transformations in the cellular residual branch. Experiments on Adamson, Norman, and Replogle demonstrate improved prediction of mean responses and cell distributions, including for unseen gene combinations. CoShare reduces Energy Distance over 2,000 highly variable genes by 44.5% on Adamson and 66.2% on Replogle relative to the best baseline for this metric on each benchmark. Ablations support the complementary contributions of mean–residual factorization and graph-routed component reuse.

1 INTRODUCTION

Predicting how genetic perturbations change gene expression is central to understanding gene function and cellular regulation (Bunne et al., 2023). Because genes and their combinations cannot be exhaustively tested, virtual cell models must learn from limited experiments to predict perturbation responses[1] unmeasured in the target cellular context.

Recent methods address different aspects of this problem. GEARS uses prior gene relationships to predict unseen perturbations (Roohani et al., 2024), while CellOT and Doloris model control and perturbed populations through optimal transport and diffusion, respectively (Bunne et al., 2023; Chi et al., 2026b). Yet with finitely many observed perturbations, multiple conditional generative models can match every training distribution while predicting different responses to the same unseen perturbation c^* (Fig. 1A, left) (D'Amour et al., 2022; Saengkyongam et al., 2024). Distributional fit alone thus leaves unseen responses underdetermined. Moreover, single-cell sequencing destroys measured cells, preventing paired control and perturbed measurements (Fig. 1A, right). We therefore predict perturbed-cell distributions rather than individual cellular trajectories.

Large-scale Perturb-seq studies report similar expression changes across perturbations (Replogle et al., 2022; Norman et al., 2019), suggesting common response components. This motivates learning shared components from measured responses and recombining them for unseen perturbations, while accounting for both average expression changes and cell-to-cell variation. For each pertur-

[1] Hereafter, "responses" is shorthand for perturbation responses.

Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.

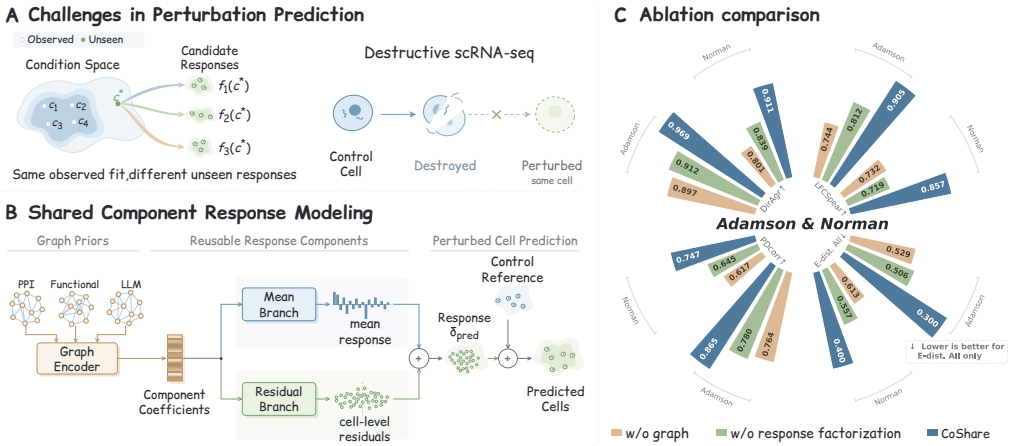

Figure 1: **Motivation, framework, and ablation results of CoShare. (A)** Models that agree on observed conditions can predict different responses to an unseen perturbation (left). Control and perturbed cells are unpaired (right). **(B) CoShare** uses component coefficients shared by both branches from gene graphs for mean prediction and residual generation. **(C)** Results in Adamson and Norman, comparing **CoShare** with variants without response factorization or graph priors.

bation and cell type, we define the response δ as each perturbed cell's expression minus the corresponding control mean. We decompose δ into a mean response, capturing the average expression change, and a cell-level residual, capturing the deviation from the perturbed population mean. Separate population mean and cellular residual branches predict the mean response and generate the residual distribution.

Factorization alone, however, does not specify how unseen perturbations reuse learned components. We introduce **CoShare**, a graph-routed factorized Delta (δ) Flow Matching framework coordinating reuse across branches (Fig. 1B). The mean branch learns shared vectors of expression change, while the residual branch learns shared hidden-feature transformations within a Flow Matching generator mapping noise to residual samples (Lipman et al., 2022). A multi-source graph encoder uses protein interactions, functional relationships, and Large Language Model(LLM) derived associations to produce a component coefficient vector. This vector controls component reuse across both branches, allowing prior gene relationships to guide how unseen perturbations recombine learned components.

Our main contributions are:

- **Mean–residual factorization.** We separate average expression changes from cell-to-cell variation through mean prediction and residual generation branches trained on unpaired control and perturbed populations.
- **Graph-routed component reuse.** We design a graph-guided router whose component coefficient vector weights shared response vectors in the mean branch and controls adapters combining shared feature transformations in the residual branch, coordinating reuse for unseen perturbations.
- **Empirical evaluation.** On the Adamson, Norman, and Replogle benchmarks with single- and double-gene perturbations, **CoShare** improves predictions of mean responses and cell distributions. It reduces Energy Distance over 2,000 highly variable genes by 44.5% on Adamson and 66.2% on Replogle relative to each benchmark's best baseline.

2 Related Work and Preliminaries

2.1 Single-Cell Perturbation Response Prediction

Single-cell perturbation prediction leverages latent representations, gene relationships, and pretraining. scGen combines variational autoencoders with latent space vector arithmetic to transfer responses across cellular contexts (Lotfollahi et al., 2019). CPA recombines disentangled

basal state, perturbation, and covariate representations to predict unseen condition combinations (Lotfollahi et al., 2023). Biolord disentangles known and unknown cellular attributes for counterfactual prediction (Piran et al., 2024). GEARS learns gene and perturbation embeddings from coexpression and Gene Ontology-derived graphs to predict unseen perturbations (Roohani et al., 2024). scGPT, scFoundation, and GeneCompass transfer pretrained single-cell representations to perturbation prediction (Cui et al., 2024; Hao et al., 2024; Yang et al., 2024).

A complementary line of work models perturbed cell distributions. CellOT learns neural optimal transport maps between unpaired control and perturbed populations (Bunne et al., 2023). PerturbDiff models distributional variability through diffusion over kernel mean embeddings of cell distributions (Yuan et al., 2026). CellFlow and scDFM use conditional Flow Matching with optimal transport pairing and maximum mean discrepancy regularization, respectively (Klein et al., 2025; Yu et al., 2026). However, fitting observed distributions alone does not determine responses to unseen perturbations. Beyond using gene relationships to learn representations or guide attention, **CoShare** uses them to coordinate shared components across condition mean prediction and cell-level residual generation. This provides a common biological basis for extrapolating both response components.

2.2 Conditional Generation and Flow Matching Preliminaries

Condition-level non-identifiability. Let y denote a condition and $\mathcal{T}$ the finite set of training conditions. We fit a conditional generative model $p_\theta(\cdot \mid y)$ to the target distribution $p^\star(\cdot \mid y)$ by minimizing

$$\mathcal{R}_{\text{train}}(\theta) = \mathbb{E}_{y\sim P_{\text{train}}}\left[\mathcal{D}(p^\star(\cdot \mid y), p_\theta(\cdot \mid y))\right], \tag{1}$$

where $\mathcal{D}$ measures distributional discrepancy and P_{train} is supported on $\mathcal{T}$.

Two models may agree on all training conditions yet predict different distributions for an unseen condition $y^* \notin \mathcal{T}$. They have the same training loss, so training fit alone does not uniquely determine predictions for unseen conditions. Extrapolation therefore requires inductive biases that relate training and unseen conditions (D'Amour et al., 2022; Saengkyongam et al., 2024).

Flow Matching with x prediction. Flow Matching learns a time-dependent velocity field that transforms a simple source distribution into a target distribution (Lipman et al., 2022). For a condition y, we sample $x \sim p^\star(\cdot \mid y)$ and $\epsilon \sim \mathcal{N}(0, I)$. A linear path connects the noise and data:

$$z_t = (1-t)\epsilon + t\,x, \qquad v^\star = x - \epsilon, \tag{2}$$

where $t \in [0, 1]$ and $v^\star$ is the path velocity used for supervision.

The x-prediction parameterization predicts the clean endpoint rather than the velocity (Li & He, 2026). Let D_θ denote the prediction network. Its output, induced velocity, and training loss are

$$\hat{x}_\theta = D_\theta(z_t, t, y), \qquad \hat{v}_\theta = \frac{\hat{x}_\theta - z_t}{1-t},\ t < 1, \qquad \mathcal{L}_{\text{FM}} = \mathbb{E}_{y,x,\epsilon,t}\left[\|\hat{v}_\theta - v^\star\|_2^2\right]. \tag{3}$$

The expectation is over training conditions, data samples, independent Gaussian noise, and times sampled from a distribution on $(0, 1)$. The network therefore predicts clean data while being trained with a velocity loss. At inference, we draw $z_0 \sim \mathcal{N}(0, I)$ and integrate the learned velocity field toward $t = 1$, holding y fixed, to generate a sample from $p_\theta(\cdot \mid y)$.

3 Method

Problem Setup. Let G denote the number of profiled genes. A *condition* (c, s) specifies genetic perturbation c and cell type s. We use c to denote a single-gene or double-gene perturbation. For each condition observed during training, $X_s^{\text{ctrl}},\ X_{c,s}^{\text{pert}} \subset \mathbb{R}^G$ denote the control and perturbed populations, respectively. Each population contains gene expression vectors from cells of type s.

Single-cell sequencing destroys the measured cell, so the control and perturbed populations are unpaired. No cell is observed both before and after perturbation. We therefore train a conditional generative model to predict the distribution of perturbed cells given the control population and (c, s).

Let $\mathcal{C}_{\text{train}}$ denote the set of genetic perturbations included in training. At inference, our goal is to generate a perturbed population for an unseen perturbation $c^* \notin \mathcal{C}_{\text{train}}$ in cell type s. The corresponding control population is available, but perturbed measurements for c^* are not. We use prior biological relationships among genes to relate c^* to training perturbations.

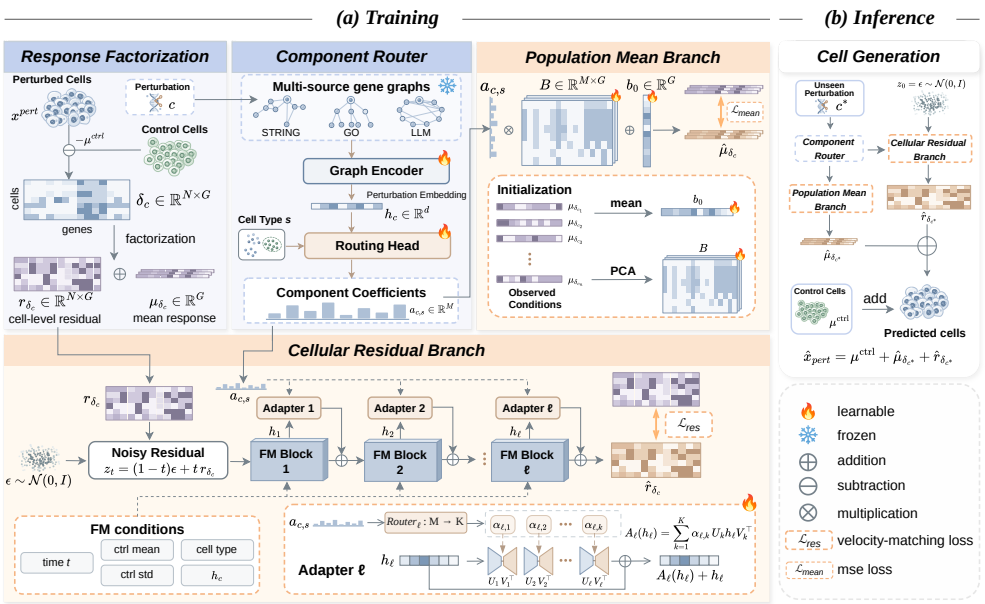

Figure 2: **Overview of CoShare. (a) Training.** Responses relative to the control mean are decomposed into mean responses and cell-level residuals. The component router uses multi-source gene graphs and cell type to produce component coefficients. These coefficients weight B in the population mean branch and control adapters in the cellular residual branch. **(b) Inference.** For an unseen perturbation, the two branches predict mean response and generate cell-level residuals from noise, respectively. Adding both to the control mean yields predicted cells.

Method Overview. CoShare (Figure 2) learns response components shared across related perturbations and uses prior gene relationships to guide their reuse for unseen perturbations. For each training condition (c, s), we subtract the control mean from each perturbed cell's expression and decompose the resulting response δ into a mean response and a cell-level residual (Section 3.1). The population mean branch predicts average expression changes, while the cellular residual branch models variation around the perturbed population mean.

A component router uses a multi-source graph encoder and a routing head to map (c, s) to component coefficients (Section 3.2). The mean branch uses these coefficients to combine shared response vectors (Section 3.3). The residual branch uses Flow Matching to generate residuals from noise, conditioned on the perturbation, cell type, and control statistics. It also maps the component coefficients to layerwise operator weights to control how shared feature transformations are combined (Section 3.4). At inference, adding the predicted mean response and generated residuals to the control mean yields a predicted population for an unseen perturbation (Section 3.5).

3.1 Mean–Residual Factorization

To focus on expression changes relative to the control state, we model perturbation responses rather than absolute perturbed expression. Let μ_s^{ctrl} denote the mean expression vector of the control population X_s^{ctrl}. For the i-th perturbed cell in condition (c, s), we define its response as

$$\delta_{c,s,i} = x_{c,s,i}^{\text{pert}} - \mu_s^{\text{ctrl}}. \tag{4}$$

The control mean μ_s^{ctrl} provides a common reference for perturbations in cell type s.

A perturbation can affect both average expression and variation among cells. We model these two aspects separately by decomposing each response into a mean response and a cell-level residual:

$$\delta_{c,s,i} = \mu_{\delta_{c,s}} + r_{\delta_{c,s}}^{(i)}, \qquad \mu_{\delta_{c,s}} = \frac{1}{N_{c,s}} \sum_{i=1}^{N_{c,s}} \delta_{c,s,i}, \tag{5}$$

where $N_{c,s}$ is the number of cells in $X^{\text{pert}}_{c,s}$. The mean response $\mu_{\delta_{c,s}}$ gives the average expression difference between the perturbed and control populations. The residual $r^{(i)}_{\delta_{c,s}}$ describes how cell i differs from the perturbed population mean. We write $r_{\delta_{c,s}}$ for a sample from the residual distribution when the cell index is omitted.

During training, we compute the mean response and residuals from the measured populations. The mean response supervises the population mean branch, while the residual samples train the cellular residual branch to model variation around that mean.

3.2 Component Coefficients for Specific Condition

We use prior gene relationships to guide how unseen perturbations reuse response components learned during training. These relationships come from three sources: protein interactions from STRING ($\mathcal{K}_{\text{STRING}}$), functional relationships from Gene Ontology ($\mathcal{K}_{\text{GO}}$), and gene associations derived from LLMs ($\mathcal{K}_{\text{LLM}}$). The graphs provide complementary information, so we use the collection $\mathcal{K} = \{\mathcal{K}_{\text{STRING}}, \mathcal{K}_{\text{GO}}, \mathcal{K}_{\text{LLM}}\}$. Graph construction is described in Appendix A.2.

A multi-source graph encoder produces a representation $h_g = f_\phi(g; \mathcal{K}) \in \mathbb{R}^d$ for each gene g. The graph connections are fixed during training, while the encoder parameters ϕ are learned. For a single-gene perturbation $c = \{g\}$, we set $h_c = h_g$. For a double-gene perturbation $c = \{g_1, g_2\}$, we combine the two gene representations as

$$h_c = \text{MLP}([h_{g_1} + h_{g_2};\ h_{g_1} \odot h_{g_2};\ |h_{g_1} - h_{g_2}|]) \in \mathbb{R}^d, \tag{6}$$

where $[\,;\,]$ denotes concatenation and $\odot$ denotes elementwise multiplication. The product and absolute difference provide information about the gene pair beyond the sum.

To allow component coefficients to vary with cell type, we concatenate h_c with a learnable cell-type embedding e_s. A routing head g_ψ maps this input to the component coefficient vector:

$$a_{c,s} = g_\psi([h_c; e_s]) \in \mathbb{R}^M. \tag{7}$$

Here, M is the number of shared response vectors in the population mean branch, and each coefficient weights one vector (Section 3.3). The cellular residual branch maps the component coefficient vector to operator weights at each layer to combine shared feature transformations (Section 3.4).

3.3 Mean Prediction with shared Response Dictionary

The population mean branch predicts the mean response for each condition by combining response components shared across perturbations. We store M basis vectors in a learnable dictionary $B \in \mathbb{R}^{M \times G}$. Each row describes a pattern of expression change across the G genes. Using the coefficients $a_{c,s}$ from Section 3.2, we predict the mean response as

$$\hat{\mu}_{\delta_{c,s}} = a_{c,s}^\top B + b_0, \tag{8}$$

where $b_0 \in \mathbb{R}^G$ is a learnable offset shared across all conditions.

Let $\mathcal{T}$ denote the set of training conditions. We use a compact dictionary with $M \ll \min(G, |\mathcal{T}|)$. For initialization, b_0 is set to the average of the mean responses across training conditions, and B to the leading M PCA directions. Both are updated during end-to-end training.

The mean branch is trained to match the empirical mean response $\mu_{\delta_{c,s}}$ from Eq. 5 by minimizing

$$\mathcal{L}_{\text{mean}} = \mathbb{E}_{(c,s)\sim\mathcal{T}} \left[\left\| \hat{\mu}_{\delta_{c,s}} - \mu_{\delta_{c,s}} \right\|_2^2 \right]. \tag{9}$$

3.4 Residual Generation with Shared Operators

Residual Flow Matching. The cellular residual branch models variation around the perturbed population mean. We employ Flow Matching to learn a time-dependent velocity field that transforms Gaussian noise into samples from the residual distribution $q(\cdot \mid c, s)$ for a given condition (c, s).

During training, we sample a residual $r_{\delta_{c,s}} \sim q(\cdot \mid c, s)$, noise $\epsilon \sim \mathcal{N}(0, I)$, and time $t \sim \text{LogitNormal}(\mu_t, \sigma_t^2)$. A linear path connects the noise to the residual:

$$z_t = (1-t)\epsilon + t\, r_{\delta_{c,s}}, \qquad v^\star = r_{\delta_{c,s}} - \epsilon, \tag{10}$$

where z_t is the network input and $v^\star$ is the target velocity. To supply cell context, the model is conditioned on $\xi_{c,s} = (\mu_s^{\text{ctrl}}, \sigma_s^{\text{ctrl}}, e_s, h_c)$, incorporating control mean μ_s^{ctrl}, control standard deviation σ_s^{ctrl}, cell-type embedding e_s, and perturbation representation h_c.

Routed Adapters. While direct conditioning $\xi_{c,s}$ provides cellular context, it leaves the reuse of shared response patterns implicit within the generator. To explicitly control component reuse, we use the graph-derived component coefficient vector $a_{c,s}$ to dynamically compose shared feature transformations via layerwise operator weights.

At each layer ℓ, we learn K low-rank operators $\{U_{\ell,k}V_{\ell,k}^\top\}_{k=1}^K$. A layer-specific router ρ_ℓ maps $a_{c,s}$ to operator weights α_ℓ:

$$\alpha_\ell = \rho_\ell(a_{c,s}) \in \mathbb{R}^K, \qquad \mathcal{A}_\ell(h_\ell) = \sum_{k=1}^{K} \alpha_{\ell,k} U_{\ell,k} V_{\ell,k}^\top h_\ell, \tag{11}$$

where h_ℓ is the hidden state. The adapter output is then added to the hidden state: $\tilde{h}_\ell = h_\ell + \mathcal{A}_\ell(h_\ell)$.

Training Objective. Biological variation in gene expression often has low-dimensional structure (Heimberg et al., 2016; Eraslan et al., 2019), whereas the isotropic Gaussian noise used in Flow Matching spans all dimensions. Recovering the clean endpoint therefore encourages the network to focus on the underlying biological signal. Motivated by this property, the generator D_θ directly predicts the clean residual (Li & He, 2026) instead of the path velocity used in standard Flow Matching. From the predicted residual endpoint $\hat{r}_\theta = D_\theta(z_t, t, \xi_{c,s}; a_{c,s})$, we compute the induced velocity:

$$\hat{v}_\theta = \frac{\hat{r}_\theta - z_t}{1 - t}. \tag{12}$$

The model is then supervised by matching this induced velocity to the target $v^\star$:

$$\mathcal{L}_{\text{res}} = \mathbb{E}_{(c,s),\, r_{\delta_{c,s}},\, \epsilon,\, t} \left[\|\hat{v}_\theta - v^\star\|_2^2 \right]. \tag{13}$$

By substituting Eq. 12 into Eq. 13, this formulation implicitly weights the squared residual prediction error by $(1-t)^{-2}$, placing greater emphasis on generating accurate residuals as $t \to 1$.

3.5 Joint Training and Extrapolation

Training. The component router and both branches are jointly optimized with

$$\mathcal{L} = \mathcal{L}_{\text{mean}} + \mathcal{L}_{\text{res}}. \tag{14}$$

Both objectives update the component router through $a_{c,s}$, coordinating how two branches combine respective shared components.

Perturbed-Cell Generation. For an unseen perturbation c^* in cell type s, the trained component router computes h_{c^*} and $a_{c^*,s}$ using the prior gene graphs and cell-type context (Section 3.2). The population mean branch then predicts $\hat{\mu}_{\delta_{c^*,s}}$ using Eq. 8. The cellular residual branch uses the same component coefficients together with $\xi_{c^*,s}$, whose control statistics come from the target cell type's control population X_s^{ctrl}.

For each generated cell, we independently draw $z_0 \sim \mathcal{N}(0, I)$ and integrate the learned velocity field $\hat{v}_\theta$ in Eq. 12 from $t = 0$ to $t = 1$. The conditioning inputs and component coefficients remain fixed during integration. The terminal state is a generated residual $\hat{r}_{\delta_{c^*,s}}$. Adding this residual to the predicted mean response and the control mean gives the perturbed expression profile:

$$\hat{x}_{c^*,s}^{\text{pert}} = \mu_s^{\text{ctrl}} + \hat{\mu}_{\delta_{c^*,s}} + \hat{r}_{\delta_{c^*,s}}. \tag{15}$$

Repeating this procedure with independent noise draws yields a predicted cell population representing $p_\theta(x \mid c^*, s)$.

4 Experiment

4.1 Experimental Setup

Baselines. We compare with nine methods for perturbation prediction: GEARS (Roohani et al., 2024), scGPT (Cui et al., 2024), scFoundation (Hao et al., 2024), GraphVCI (Wu et al., 2022),

Table 1: Results on held-out perturbations in Adamson and Replogle. Gra., Fou., and Gen. denote graph-based, foundation, and generative methods.

Type	Method	RMSE All ↓	E-dist. All ↓	RMSE DE40 ↓	E-dist. DE40 ↓	DirAgr↑	LFCSpear↑	PDCorr↑	PDS_{cos}↑
Adamson									
Gra.	GEARS (Roohani et al., 2024)	0.048	0.541	0.138	0.554	0.903	0.841	0.673	0.537
Gra.	GraphVCI (Wu et al., 2022)	0.050	2.841	0.137	1.200	0.893	0.786	0.655	0.523
Fou.	scGPT (Cui et al., 2024)	0.053	2.498	0.111	0.987	0.911	0.812	0.669	0.607
Fou.	scFoundation (Hao et al., 2024)	0.101	2.407	0.174	1.053	0.794	0.769	0.381	0.576
Gen.	Departures (Chi et al., 2026a)	0.050	0.556	0.109	0.460	0.889	0.755	0.656	**0.653**
Gen.	Doloris (Chi et al., 2026b)	0.068	0.668	0.150	0.595	0.781	0.571	0.570	0.521
Gen.	scDFM (Yu et al., 2026)	0.090	1.178	0.175	0.765	0.820	0.769	0.486	0.506
Gen.	scBIG (Ruan et al., 2026)	0.037	1.784	0.103	0.680	0.954	0.862	0.815	0.535
Gen.	PerturbDiff (Yuan et al., 2026)	0.040	0.674	0.108	0.552	0.926	0.825	0.794	0.562
	CoShare (Ours)	**0.032**	**0.300**	**0.091**	**0.362**	**0.969**	**0.905**	**0.865**	0.616
Replogle									
Gra.	GEARS (Roohani et al., 2024)	0.086	0.648	0.218	0.773	0.773	0.486	0.398	0.534
Gra.	GraphVCI (Wu et al., 2022)	0.512	5.760	0.773	2.652	0.579	0.151	0.136	0.501
Fou.	scGPT (Cui et al., 2024)	0.086	2.828	0.209	1.313	0.784	0.569	0.403	0.529
Fou.	scFoundation (Hao et al., 2024)	0.148	2.632	0.267	1.325	0.631	0.235	0.309	0.546
Gen.	Departures (Chi et al., 2026a)	0.117	0.975	0.289	0.949	0.565	0.262	0.154	0.512
Gen.	Doloris (Chi et al., 2026b)	0.096	0.725	0.229	0.783	0.633	0.251	0.278	0.509
Gen.	scDFM (Yu et al., 2026)	0.151	1.677	0.285	1.038	0.666	0.260	0.242	0.502
Gen.	scBIG (Ruan et al., 2026)	0.075	1.715	0.166	0.753	0.805	0.587	0.466	0.591
Gen.	PerturbDiff (Yuan et al., 2026)	0.103	0.816	0.219	0.765	0.695	0.359	0.318	0.511
	CoShare (Ours)	**0.068**	**0.219**	**0.154**	**0.521**	**0.857**	**0.629**	**0.517**	**0.770**

Table 2: Performance on Norman double-gene perturbations across 0/2, 1/2, and 2/2 unseen gene.

Type	Method	0/2 Unseen		1/2 Unseen		2/2 Unseen	
		E-dist. DE40 ↓	PDCorr ↑	E-dist. DE40 ↓	PDCorr ↑	E-dist. DE40 ↓	PDCorr ↑
Gra.	GEARS (Roohani et al., 2024)	0.854	0.504	0.885	0.502	0.853	0.500
Gra.	GraphVCI (Wu et al., 2022)	1.465	0.396	1.399	0.412	1.271	0.463
Fou.	scGPT (Cui et al., 2024)	1.064	0.829	1.114	0.658	1.153	0.508
Fou.	scFoundation (Hao et al., 2024)	1.118	0.421	1.137	0.435	1.097	0.450
Gen.	Departures (Chi et al., 2026a)	0.823	0.320	0.870	0.314	0.764	0.288
Gen.	Doloris (Chi et al., 2026b)	0.698	0.475	0.771	0.421	0.776	0.421
Gen.	scDFM (Yu et al., 2026)	0.687	0.679	0.778	0.552	0.887	0.497
Gen.	scBIG (Ruan et al., 2026)	0.775	0.755	0.791	0.689	0.862	0.608
Gen.	PerturbDiff (Yuan et al., 2026)	0.530	0.806	0.614	0.726	0.647	0.645
	CoShare (Ours)	**0.240**	**0.849**	**0.349**	**0.760**	**0.299**	**0.745**

Departures (Chi et al., 2026a), Doloris (Chi et al., 2026b), scDFM (Yu et al., 2026), scBIG (Ruan et al., 2026), and PerturbDiff (Yuan et al., 2026). These baselines cover methods using gene graphs, foundation models, causal modeling, and generative modeling of cell populations. All methods use same data splits and evaluation protocol. Details are provided in Appendix A.3.

Data. We evaluate on three genetic Perturb-seq benchmarks. Adamson (Adamson et al., 2016) contains single-gene CRISPRi perturbations in K562 cells. Norman (Norman et al., 2019) profiles single- and double-gene CRISPRa perturbations in K562 cells. Replogle (Replogle et al., 2022; Nadig et al., 2025) comprises CRISPRi screens across K562, RPE1, Jurkat, and HepG2 cells. Dataset details, preprocessing, and data splits are provided in Appendix A.1.1.

Metrics. We evaluate predictions from complementary perspectives of expression fidelity and perturbation-effect recovery. Following prior perturbation benchmarks (Chi et al., 2026b), we report RMSE and Energy Distance (E-dist.) over all genes and the top 40 differentially expressed genes (DE40), measuring population-level accuracy and distributional agreement. We further adopt four perturbation-aware metrics from Cell-Eval (Adduri et al.; Yuan et al., 2026): Direction Agreement (DirAgr), LFC Spearman correlation (LFCSpear), Pearson Delta Correlation (PDCorr), and cosine-based Perturbation Discrimination Score (PDS_{cos}), which assess the direction, ranking, correlation, and specificity of predicted perturbation effects, respectively. Lower is better for RMSE and E-dist.; higher is better for the remaining metrics. Definitions are provided in Appendix A.1.2.

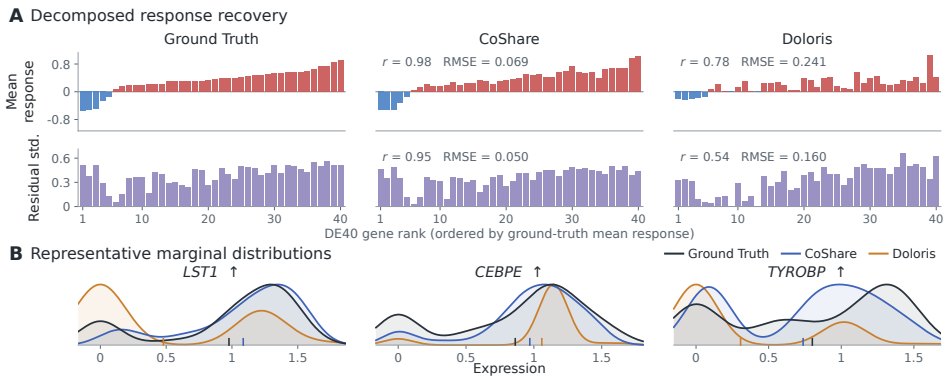

Figure 3: **Response recovery for the held-out `CEBPB+CEBPE` combination in Norman. (A)** Ground-truth and predicted mean responses relative to the respective controls (top) and residual standard deviations across cells (bottom) for DE40 genes. **(B)** Peak-normalized marginal expression distributions for three representative DE genes. Arrows indicate positive ground-truth condition means; vertical marks denote population means.

4.2 Generalization to Unseen Perturbations

Table 1 compares predictions for perturbations held out in the target cell type. On Adamson, **CoShare** achieves the best results on seven of the eight metrics. Its largest improvement is in Energy Distance over all genes: E-dist. All decreases from 0.541 for GEARS, the best baseline, to 0.300, a 44.5% reduction. **CoShare** achieves the lowest RMSE on both All genes and DE40, and increases PDCorr from the best baseline value of 0.815 for scBIG to 0.865. These results show improved prediction of both average expression changes and expression distributions across cells.

On Replogle, **CoShare** ranks first on all eight metrics for the held-out HepG2 perturbations. E-dist. All decreases from 0.648 for GEARS to 0.219, a 66.2% reduction. **CoShare** also improves PDCorr from 0.466 to 0.517 and $\text{PDS}_{\cos}$ from 0.591 to 0.770 relative to scBIG, the strongest baseline on both metrics. The higher $\text{PDS}_{\cos}$ indicates that predictions better distinguish the corresponding measured response from responses to other test perturbations. **CoShare** therefore improves distributional accuracy while preserving the differences between perturbation responses.

4.3 Compositional Generalization

We evaluate predictions for unseen perturbations and gene combinations on Norman. On the complete test set, **CoShare** achieves the best results on all metrics (Appendix A.4). Relative to the best baseline, E-dist. DE40 decreases from 0.611 to 0.375, and PDCorr increases from 0.669 to 0.747. To examine generalization to new combinations, we group held-out gene pairs by how many genes were unseen as perturbation targets during training (Table 2): 0/2, 1/2, or 2/2; each combination itself is unseen. **CoShare** achieves the lowest E-dist. DE40 and highest PDCorr in all three groups. In the 2/2 unseen group, it reduces E-dist. DE40 from 0.647 to 0.299, a 53.8% reduction, and increases PDCorr from 0.645 to 0.745 over PerturbDiff, the best baseline on both metrics. These gains extend to combinations where both genes were unseen during training.

Figure 3 illustrates response recovery for the held-out `CEBPB+CEBPE` combination in Norman. Panel A compares mean responses and residual standard deviations across cells for the DE40 genes. **CoShare** achieves higher Pearson correlations than Doloris for mean responses (0.98 vs. 0.78) and residual variability (0.95 vs. 0.54), with lower RMSE for both. Panel B shows closer agreement with the measured expression distributions of *LST1*, *CEBPE*, and *TYROBP*. Together, these results illustrate recovery of both average expression changes and cellular variation for unseen perturbation.

4.4 Ablation

Contributions of graph guidance, factorization, and routed adapters. Figure 4 compares **CoShare** with Base Flow Matching (FM) and three ablated variants on Norman. **CoShare** per-

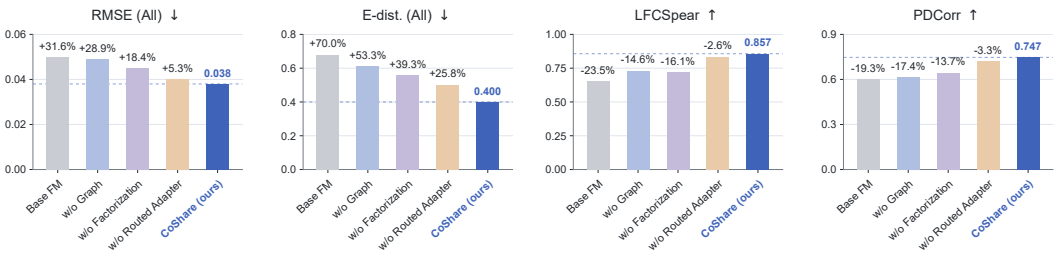

Figure 4: Ablation of graph guidance, factorization, and routed adapter on Norman

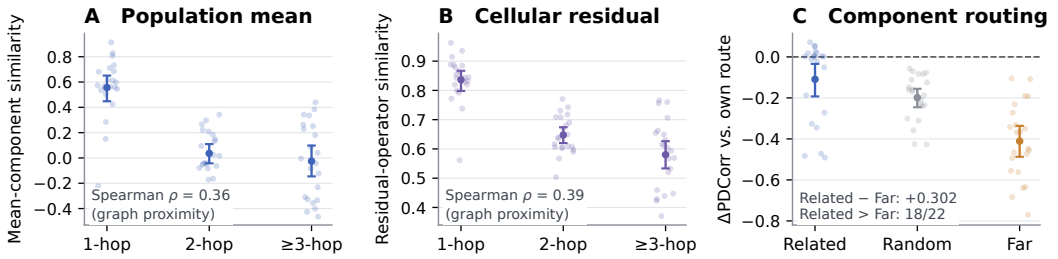

Figure 5: **(A)** Similarity of mean branch component coefficients and **(B)** residual operators between held-out and training perturbations, grouped by perturbed gene distance in the biological graph. **(C)** PDCorr change when a held-out perturbation uses coefficients from a related, random, or distant training perturbation instead of its own.

forms best on all four metrics, reducing E-dist. All from 0.680 to 0.400 and increasing PDCorr from 0.603 to 0.747 over Base FM.

Removing graph guidance raises E-dist. All to 0.613 and lowers PDCorr to 0.617, while removing factorization or routed adapters increases E-dist. All to 0.557 and 0.503, respectively. All three ablations degrade the four metrics, showing that these components provide complementary benefits for perturbation prediction.

Perturbations of nearby genes combine shared components similarly. On Adamson, we compare **CoShare**'s component coefficients in the mean branch and effective residual operators between held-out and training perturbations. Both show higher similarity when the target genes are closer in the biological graph (Fig. 5A–B). Details are provided in Appendix A.1.3.

We further replace each test perturbation's component coefficients with those from a training perturbation of a nearby, random, or distant gene, keeping all other inputs fixed (Fig. 5C). Relative to using its own coefficients, replacements from nearby genes cause smaller average drops in PDCorr than random or distant replacements. They outperform distant replacements by 0.302 on average and in 18 of 22 test perturbations. Together, these results support using graph proximity to guide the reuse of learned response components.

5 Conclusion

We presented **CoShare**, a graph-routed factorized Delta Flow Matching framework for predicting responses to unseen genetic perturbations. CoShare decomposes responses relative to the control mean into mean responses and cell-level residuals. Graph-derived component coefficients coordinate the reuse of shared response vectors in the mean branch and shared feature transformations in the residual branch. Experiments on Adamson, Norman, and Replogle demonstrate improved prediction of mean responses and cell distributions, including for unseen gene combinations. Ablations support the complementary contributions of mean–residual factorization and graph-routed component reuse, while routing analyses show that perturbations of related genes use shared components similarly. Together, these results suggest that learning shared response components and guiding their reuse with prior gene relationships can improve extrapolation to unseen perturbations.

AI USE STATEMENT

Large language models (LLMs) were used to construct the gene association graph. Apart from this use, LLMs were used solely for general-purpose assistance with text refinement and language polishing. The research ideas, methodology, experimental design, analysis, and scientific conclusions presented in this paper were developed by the authors.

REFERENCES

Britt Adamson, Thomas M Norman, Marco Jost, Min Y Cho, James K Nuñez, Yuwen Chen, Jacqueline E Villalta, Luke A Gilbert, Max A Horlbeck, Marco Y Hein, et al. A multiplexed single-cell crispr screening platform enables systematic dissection of the unfolded protein response. *Cell*, 167(7):1867–1882, 2016.

Abhinav K Adduri, Dhruv Gautam, Beatrice Bevilacqua, Mohsen Naghipourfar, Alishba Imran, Rohan Shah, Noam Teyssier, Rishi Verma, Christopher Carpenter, Basak Eraslan, et al. Predicting cellular responses to perturbation across diverse contexts with state. *Cell*.

Charlotte Bunne, Stefan G Stark, Gabriele Gut, Jacobo Sarabia Del Castillo, Mitch Levesque, Kjong-Van Lehmann, Lucas Pelkmans, Andreas Krause, and Gunnar Rätsch. Learning single-cell perturbation responses using neural optimal transport. *Nature methods*, 20(11):1759–1768, 2023.

Changxi Chi, Yufei Huang, Jun Xia, Jiangbin Zheng, Yunfan Liu, Zelin Zang, and Stan Z Li. Departures: Distributional transport for single-cell perturbation prediction with neural schrödinger bridges. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 40, pp. 20544–20552, 2026a.

Changxi Chi, Jun Xia, Yufei Huang, Zhuoli Ouyang, Cheng Tan, Yunfan Liu, Jingbo Zhou, Chang Yu, Liangyu Yuan, Siyuan Li, et al. Doloris: Dual conditional diffusion implicit bridges with sparsity masking strategy for unpaired single-cell perturbation estimation. In *International Conference on Learning Representations*, volume 2026, pp. 52808–52823, 2026b.

Haotian Cui, Chloe Wang, Hassaan Maan, Kuan Pang, Fengning Luo, Nan Duan, and Bo Wang. scgpt: toward building a foundation model for single-cell multi-omics using generative ai. *Nature methods*, 21(8):1470–1480, 2024.

Alexander D'Amour, Katherine Heller, Dan Moldovan, Ben Adlam, Babak Alipanahi, Alex Beutel, Christina Chen, Jonathan Deaton, Jacob Eisenstein, Matthew D Hoffman, et al. Underspecification presents challenges for credibility in modern machine learning. *Journal of Machine Learning Research*, 23(226):1–61, 2022.

Gökcen Eraslan, Lukas M Simon, Maria Mircea, Nikola S Mueller, and Fabian J Theis. Single-cell rna-seq denoising using a deep count autoencoder. *Nature communications*, 10(1):390, 2019.

Minsheng Hao, Jing Gong, Xin Zeng, Chiming Liu, Yucheng Guo, Xingyi Cheng, Taifeng Wang, Jianzhu Ma, Xuegong Zhang, and Le Song. Large-scale foundation model on single-cell transcriptomics. *Nature methods*, 21(8):1481–1491, 2024.

Graham Heimberg, Rajat Bhatnagar, Hana El-Samad, and Matt Thomson. Low dimensionality in gene expression data enables the accurate extraction of transcriptional programs from shallow sequencing. *Cell systems*, 2(4):239–250, 2016.

Dominik Klein, Jonas Simon Fleck, Daniil Bobrovskiy, Lea Zimmermann, Sören Becker, Alessandro Palma, Leander Dony, Alejandro Tejada-Lapuerta, Guillaume Huguet, Hsiu-Chuan Lin, et al. Cellflow enables generative single-cell phenotype modeling with flow matching. *bioRxiv*, 2025.

Tianhong Li and Kaiming He. Back to basics: Let denoising generative models denoise. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 36115–36125, 2026.

Yaron Lipman, Ricky TQ Chen, Heli Ben-Hamu, Maximilian Nickel, and Matt Le. Flow matching for generative modeling. *arXiv preprint arXiv:2210.02747*, 2022.

Mohammad Lotfollahi, F Alexander Wolf, and Fabian J Theis. scgen predicts single-cell perturbation responses. *Nature methods*, 16(8):715–721, 2019.

Mohammad Lotfollahi, Anna Klimovskaia Susmelj, Carlo De Donno, Leon Hetzel, Yuge Ji, Ignacio L Ibarra, Sanjay R Srivatsan, Mohsen Naghipourfar, Riza M Daza, Beth Martin, et al. Predicting cellular responses to complex perturbations in high-throughput screens. *Molecular systems biology*, 19(6):MSB202211517, 2023.

Ajay Nadig, Joseph M Replogle, Angela N Pogson, Mukundh Murthy, Steven A McCarroll, Jonathan S Weissman, Elise B Robinson, and Luke J O'Connor. Transcriptome-wide analysis of differential expression in perturbation atlases. *Nature Genetics*, 57(5):1228–1237, 2025.

Thomas M Norman, Max A Horlbeck, Joseph M Replogle, Alex Y Ge, Albert Xu, Marco Jost, Luke A Gilbert, and Jonathan S Weissman. Exploring genetic interaction manifolds constructed from rich single-cell phenotypes. *Science*, 365(6455):786–793, 2019.

Zoe Piran, Niv Cohen, Yedid Hoshen, and Mor Nitzan. Disentanglement of single-cell data with biolord. *Nature Biotechnology*, 42(11):1678–1683, 2024.

Joseph M Replogle, Reuben A Saunders, Angela N Pogson, Jeffrey A Hussmann, Alexander Lenail, Alina Guna, Lauren Mascibroda, Eric J Wagner, Karen Adelman, Gila Lithwick-Yanai, et al. Mapping information-rich genotype-phenotype landscapes with genome-scale perturb-seq. *Cell*, 185(14):2559–2575, 2022.

Yusuf Roohani, Kexin Huang, and Jure Leskovec. Predicting transcriptional outcomes of novel multigene perturbations with gears. *Nature biotechnology*, 42(6):927–935, 2024.

Jiafa Ruan, Ruijie Quan, Liyang Xu, Zongxin Yang, and Yi Yang. Beyond independent genes: Learning module-inductive representations for single-cell gene perturbation prediction. *arXiv preprint arXiv:2602.04901*, 2026.

Sorawit James Saengkyongam, Elan Rosenfeld, Pradeep K Ravikumar, Niklas Pfister, and Jonas Peters. Identifying representations for intervention extrapolation. In *International Conference on Learning Representations*, volume 2024, pp. 39256–39281, 2024.

Yulun Wu, Robert A Barton, Zichen Wang, Vassilis N Ioannidis, Carlo De Donno, Layne C Price, Luis F Voloch, and George Karypis. Predicting cellular responses with variational causal inference and refined relational information. *arXiv preprint arXiv:2210.00116*, 2022.

Xiaodong Yang, Guole Liu, Guihai Feng, Dechao Bu, Pengfei Wang, Jie Jiang, Shubai Chen, Qinmeng Yang, Hefan Miao, Yiyang Zhang, et al. Genecompass: deciphering universal gene regulatory mechanisms with a knowledge-informed cross-species foundation model. *Cell Research*, 34 (12):830–845, 2024.

Chenglei Yu, Chuanrui Wang, Bangyan Liao, and Tailin Wu. scdfm: Distributional flow matching model for robust single-cell perturbation prediction. *arXiv preprint arXiv:2602.07103*, 2026.

Xinyu Yuan, Xixian Liu, Ya Shi Zhang, Zuobai Zhang, Hongyu Guo, and Jian Tang. Perturbdiff: Functional diffusion for single-cell perturbation modeling. *arXiv preprint arXiv:2602.19685*, 2026.

A Appendix

A.1 Additional Experimental Details

A.1.1 Dataset Details

For each benchmark, we use the top 2,000 highly variable genes (HVGs), identified using Scanpy's `sc.pp.highly_variable_genes` function, as the expression space.

Adamson. The Adamson benchmark (Adamson et al., 2016) contains single-gene CRISPRi perturbations in K562 cells. The processed data contain 87 perturbations and a control population. We randomly assign 75% of the perturbations to training and 25% to testing. All cells from a held-out perturbation are assigned to the test set.

Norman. The Norman benchmark (Norman et al., 2019) contains single- and double-gene CRISPRa perturbations in K562 cells. It includes 105 single-gene and 131 double-gene perturbations, giving 236 non-control conditions. We use the GEARS *simulation* split (Roohani et al., 2024) to evaluate generalization to unseen perturbations and gene combinations. Held-out gene pairs are grouped by how many constituent genes were unseen as perturbation targets during training: **0/2 unseen** means that both genes were seen, **1/2 unseen** that one was unseen, and **2/2 unseen** that both were unseen. Each test combination itself is unseen. The test set also includes held-out single-gene perturbations.

Replogle. The Replogle benchmark combines the K562 and RPE1 CRISPRi screens from Replogle et al. (2022) with the Jurkat and HepG2 screens from Nadig et al. (2025). The processed data contain 2,023 distinct perturbations across the four cell lines. Training uses K562, RPE1, and Jurkat data together with a randomly sampled 30% of HepG2 perturbations. The remaining 70% of HepG2 perturbations are held out for evaluation. Training and test perturbations are disjoint within HepG2.

A.1.2 Evaluation Metrics

We evaluate perturbations within the target cell type and omit its index s below. Let $\mathcal{C}_{\text{test}}$ denote the set of test perturbations and $C = |\mathcal{C}_{\text{test}}|$. For perturbation c, let $X_c^{\text{pert}} = \{x_{c,i}^{\text{pert}}\}_{i=1}^{n_c}$ and $\widehat{X}_c^{\text{pert}} = \{\hat{x}_{c,j}^{\text{pert}}\}_{j=1}^{m_c}$ denote the observed and predicted perturbed cell populations. We compute a score for each test perturbation and macro-average these scores over $\mathcal{C}_{\text{test}}$.

RMSE. Following Chi et al. (2026b), RMSE compares the mean expression profiles of the predicted and observed populations. Let $\mu_{c,g}$ and $\hat{\mu}_{c,g}$ denote the observed and predicted mean expression of gene g. For the gene set $\mathcal{G}_c$ evaluated under perturbation c, we report

$$\text{RMSE} = \frac{1}{C} \sum_{c \in \mathcal{C}_{\text{test}}} \sqrt{\frac{1}{|\mathcal{G}_c|} \sum_{g \in \mathcal{G}_c} \left(\hat{\mu}_{c,g} - \mu_{c,g}\right)^2}. \tag{16}$$

For **All**, $\mathcal{G}_c$ contains all 2,000 HVGs. For **DE40**, it contains the top 40 differentially expressed genes for condition c, as defined below.

Energy Distance. We use the empirical Energy Distance from Chi et al. (2026b). For one perturbation, we omit the condition index and the superscript pert. With n observed cells and m predicted cells, the distance is

$$\begin{aligned} D_{\text{E}}(X, \widehat{X}) = \frac{2}{nm} \sum_{i=1}^{n} \sum_{j=1}^{m} \|x_i - \hat{x}_j\|_2 \\ - \frac{1}{n^2} \sum_{i=1}^{n} \sum_{i'=1}^{n} \|x_i - x_{i'}\|_2 \\ - \frac{1}{m^2} \sum_{j=1}^{m} \sum_{j'=1}^{m} \|\hat{x}_j - \hat{x}_{j'}\|_2 . \end{aligned} \tag{17}$$

The reported E-dist. is the macro-average of $D_{\mathrm{E}}(X_c^{\mathrm{pert}}, \widehat{X}_c^{\mathrm{pert}})$ over test perturbations. We evaluate it over both **All** genes and the condition-specific **DE40** subset.

DE40. For each perturbation, DE40 contains the top 40 genes ranked by differential-expression analysis between the observed perturbed and control populations. This ground-truth gene set is fixed across methods and is used for both RMSE DE40 and E-dist. DE40.

Perturbation-aware metrics. We use Cell-Eval v0.6.6 (Adduri et al.; Yuan et al., 2026) to compute DirAgr, LFCSpear, PDCorr, and $\mathrm{PDS}_{\cos}$. Cell-Eval forms pseudobulk expression vectors for the perturbed and control populations. Let μ_c and $\widehat{\mu}_c$ denote the observed and predicted mean expression vectors. The corresponding mean responses are

$$\mu_{\delta_c} = \mu_c - \mu^{\mathrm{ctrl}}, \qquad \mu_{\delta_c}^{\mathrm{pred}} = \widehat{\mu}_c - \widehat{\mu}^{\mathrm{ctrl}}, \tag{18}$$

where μ^{ctrl} and $\widehat{\mu}^{\mathrm{ctrl}}$ are the corresponding control means. The superscript pred denotes the mean response computed from predicted cells, distinguishing it from the population mean branch output.

Pearson Delta Correlation (PDCorr). For each perturbation, PDCorr is the Pearson correlation between the predicted and observed mean response vectors:

$$\mathrm{PDCorr}_c = \rho_{\mathrm{Pearson}}\left(\mu_{\delta_c}^{\mathrm{pred}}, \mu_{\delta_c}\right). \tag{19}$$

Perturbation Discrimination Score ($\mathrm{PDS}_{\cos}$). $\mathrm{PDS}_{\cos}$ measures how well a predicted mean response distinguishes the corresponding observed response from those of other test perturbations. For each test perturbation c, Cell-Eval computes the cosine distance between its predicted mean response and every observed test mean response:

$$d_{c,k} = 1 - \frac{(\mu_{\delta_c}^{\mathrm{pred}})^\top \mu_{\delta_k}}{\|\mu_{\delta_c}^{\mathrm{pred}}\|_2 \|\mu_{\delta_k}\|_2}. \tag{20}$$

Let $r_c \in \{0, \ldots, C-1\}$ be the zero-based rank of $d_{c,c}$ after sorting $\{d_{c,k}\}_{k \in \mathcal{C}_{\mathrm{test}}}$ in ascending order. Then

$$\mathrm{PDS}_{\cos,c} = 1 - \frac{r_c}{C}, \qquad \mathrm{PDS}_{\cos} = \frac{1}{C} \sum_{c \in \mathcal{C}_{\mathrm{test}}} \mathrm{PDS}_{\cos,c}\,. \tag{21}$$

For expression-space evaluation, Cell-Eval excludes the directly targeted gene from the similarity calculation when it can be matched to the perturbation label.

Direction Agreement (DirAgr) and LFC Spearman correlation (LFCSpear). Cell-Eval performs differential-expression analysis separately on the observed and predicted populations, each relative to its corresponding control population. Let

$$\mathcal{S}_c = \left\{ g : \mathrm{FDR}_{c,g}^{\mathrm{real}} < 0.05 \right\} \tag{22}$$

denote the significant ground-truth DE genes for perturbation c. Direction Agreement is

$$\mathrm{DirAgr}_c = \frac{1}{|\mathcal{S}_c|} \sum_{g \in \mathcal{S}_c} \mathbb{I}\left[\mathrm{sgn}\left(\widehat{\mathrm{LFC}}_{c,g}\right) = \mathrm{sgn}\left(\mathrm{LFC}_{c,g}\right) \right], \tag{23}$$

and LFCSpear is

$$\mathrm{LFCSpear}_c = \rho_{\mathrm{Spearman}}\left(\{\widehat{\mathrm{LFC}}_{c,g}\}_{g \in \mathcal{S}_c}, \{\mathrm{LFC}_{c,g}\}_{g \in \mathcal{S}_c} \right). \tag{24}$$

Cell-Eval v0.6.6 uses its default Wilcoxon rank-sum procedure and Benjamini–Hochberg correction to obtain FDR-adjusted p-values. We macro-average the DirAgr and LFCSpear scores over test perturbations.

A.1.3 Component Coefficient and Residual Operator Similarity

On Adamson, Fig. 5A–B compares component coefficients in the population mean branch and effective operators in the cellular residual branch. Both comparisons use the component coefficient vector $a_{c,s}$ together with the shared components in the corresponding branch. We compare held-out–training perturbation pairs (c, c') within the same cell type s and group them by the graph distance between their target genes.

Table 3: Prior gene graphs used by **CoShare**. Edge counts refer to undirected edges before each edge is represented in both directions.

Source	Edges	Retention rule
STRING	79,892	score ≥ 700, top 50
GO–BP	130,232	top 30, degree cap 60
GO–MF	115,595	top 30, degree cap 60
GO–CC	99,479	top 30, degree cap 60
LLM	219,263	top 30, degree cap 80

Component coefficient similarity (Fig. 5A). Let $B_{m,:}$ be the m-th shared response vector in the dictionary B. To account for its magnitude, we scale each component coefficient by the norm of the corresponding vector and compute cosine similarity:

$$q^{\mathrm{mean}}_{c,s,m} = a_{c,s,m}\|B_{m,:}\|_2, \qquad S_{\mathrm{mean}}(c,c';s) = \cos\big(q^{\mathrm{mean}}_{c,s}, q^{\mathrm{mean}}_{c',s}\big)\,. \tag{25}$$

Here $q^{\mathrm{mean}}_{c,s} \in \mathbb{R}^M$ is the scaled component coefficient vector, and cos denotes cosine similarity. The scaling preserves each coefficient's sign. This comparison measures similarity in the use of shared response vectors, accounting for their norms; it does not compare the predicted mean responses directly.

Residual operator similarity (Fig. 5B). At each layer ℓ with a routed adapter, the layer-specific router ρ_ℓ maps $a_{c,s}$ to operator weights $\alpha_\ell(c,s) = \rho_\ell(a_{c,s})$. The effective residual operator has the matrix representation

$$\mathbf{A}_\ell(c,s) = \sum_{k=1}^{K} \alpha_{\ell,k}(c,s) U_{\ell,k} V_{\ell,k}^\top. \tag{26}$$

This matrix gives the adapter transformation in the main text: $\mathcal{A}_\ell(h_\ell) = \mathbf{A}_\ell(c,s)h_\ell$. We compute Frobenius cosine similarity between the two operators and average over the L layers with routed adapters:

$$S_{\mathrm{res}}(c,c';s) = \frac{1}{L}\sum_{\ell=1}^{L} \frac{\langle \mathbf{A}_\ell(c,s), \mathbf{A}_\ell(c',s)\rangle_F}{\|\mathbf{A}_\ell(c,s)\|_F \|\mathbf{A}_\ell(c',s)\|_F}. \tag{27}$$

Here $\langle\cdot,\cdot\rangle_F$ is the Frobenius inner product. The comparison includes both the operator weights and the learned low-rank operators, capturing the effective feature transformations rather than the operator weights alone.

A.2 Graph Construction and Component Router

A.2.1 Prior Graphs and Preprocessing

CoShare uses prior gene relationships from three sources: STRING protein interactions, functional relationships from Gene Ontology (GO) annotations, and LLM-derived gene associations. GO provides three graph views corresponding to biological process (BP), molecular function (MF), and cellular component (CC). Together with STRING and LLM, these give five graph views over the same 10,707 genes, identified by gene symbols. All graphs are constructed before training, and their connections remain fixed. We remove self-loops and represent each retained undirected edge in both directions with the same edge weight. Table 3 summarizes the prior gene graphs.

For the GO views, we exclude terms assigned to fewer than 3 or more than 300 genes. The LLM graph is constructed from stored 3,072-dimensional gene embeddings. The saved graph metadata list `gpt-5.4-mini` as the text model and `text-embedding-3-large` as the embedding model. We align graph nodes with expression genes by gene symbol; all 2,000 expression genes in each benchmark have valid graph-node mappings. The stored node features are centered, L2-normalized, projected to 512 dimensions, and augmented with a learnable node embedding. Edge weights enter the attention logits through a logarithmic prior.

A.2.2 Multi-Source Graph Encoding and Component Routing

Each graph view is processed by a separate one-layer edge-attention encoder with width $d = 512$. Encoder parameters are not shared across views. Each encoder contains a $512 \to 1024 \to 512$ GELU feed-forward network and uses dropout 0.1. To combine the five representations for each gene, a shared scoring network produces one logit per graph view. A softmax across views gives gene-specific graph-view weights. The weighted representation is combined with the projected base node feature and passed through a fusion MLP. The graph encoder and fusion parameters are trained jointly with the rest of **CoShare**.

For a single-gene perturbation, h_c is the corresponding fused gene representation h_g. For a double-gene perturbation, the symmetric features in Eq. 6 are mapped to h_c by a $1536 \to 1024 \to 512$ MLP with GELU, dropout 0.1, and LayerNorm. This representation is invariant to the order of the two target genes.

The routing head g_ψ in Eq. 7 maps the concatenation of h_c and the cell-type embedding e_s to the component coefficient vector $a_{c,s} \in \mathbb{R}^M$. We use $M = 16$ for Adamson and Norman and $M = 64$ for Replogle. The coefficients are not constrained to be positive or to lie on a simplex. In the cellular residual branch, h_c and e_s enter the direct conditioning $\xi_{c,s}$, while layer-specific routers map the same $a_{c,s}$ to operator weights that combine shared feature transformations.

A.3 Baseline Implementations and Dataset Adaptation

We adapt the public implementations of all nine baselines to Adamson, Norman, and Replogle. For each benchmark, the dataset interface loads the expression matrix and specified data splits, maps gene and perturbation identifiers to the model's input format, and aligns predictions to the benchmark gene order. Below, we describe these interfaces and the model components used for each baseline.

GEARS. GEARS uses gene coexpression and perturbation-similarity graphs to predict responses from control cells and perturbation identities. We map the benchmark AnnData annotations to its gene, condition, and cell-type fields and load the specified data splits through its custom-split interface. A single-gene perturbation is encoded by its target gene and a control token; a double-gene perturbation retains both targets. Graph indices and expression-dependent metadata are aligned with the benchmark expression genes. Targets absent from the GO annotations remain as isolated graph nodes, preserving their perturbation conditions.

scGPT. We use the perturbation-prediction `TransformerGenerator`, initializing its gene, value, and Transformer encoders from a pretrained checkpoint. The dataset interface supplies control expression values, gene tokens, and perturbation flags. Gene symbols are mapped to the pretrained vocabulary using available aliases. Targets absent from the selected expression tokens are added as input-only tokens with zero expression and active perturbation flags. These tokens do not add expression outputs or evaluation genes.

scFoundation. The scFoundation baseline combines a frozen pretrained gene encoder with a trainable GEARS perturbation predictor. We map benchmark control expression profiles to the encoder's reference gene vocabulary using gene symbols and available identifier mappings, filling unobserved input entries with zeros. The resulting gene representations are selected in benchmark gene order and passed to the GEARS prediction network. Perturbation identities, including both targets of Norman gene pairs, use the GEARS input format.

GraphVCI. GraphVCI combines variational causal inference with a gene graph to model counterfactual expression. We map the benchmark annotations to its perturbation, control, dosage, cell-type, and split fields. Each genetic perturbation target receives unit dosage, including both genes in a Norman pair. The specified validation and held-out test sets are mapped to its native `test` and `ood` labels, respectively. We construct the relational graph by restricting the supplied STRING edges to the expression vocabulary, removing duplicate edges, and representing each undirected interaction in both directions with unit edge weights.

Departures. Departures learns conditional transport between control and perturbed cell populations. We retain its neural Schrödinger-bridge model and optimal-transport resampling while adapting the dataset interface to the benchmark AnnData objects and data splits. The gene graph combines coexpression edges with semantic neighbors from pretrained scGPT gene embeddings. For Replogle, targets absent from the expression gene set are retained as condition-only graph nodes with zero-padded expression. These nodes are excluded from expression losses and evaluation.

Doloris. Doloris combines conditional diffusion with a separate masking model for sparse expression profiles. We adapt its genetic-perturbation data loader to the benchmark expression genes and specified data splits, retaining the gene-graph conditioning and masking pathway. Adamson provides single-gene perturbations. For Norman gene pairs, both targets are supplied to the combination-aware prediction and masking interfaces. The Replogle interface uses cell-type annotations and a dedicated perturbation vocabulary, including for targets absent from the expression outputs. The diffusion and masking models use the same condition identifiers.

scDFM. scDFM combines conditional Flow Matching with a distribution-matching loss and graph-informed gene representations. We use the specified benchmark data splits and retain the coexpression-mask and perturbation-conditioning interfaces. Gene tokens are constructed from the expression vocabulary, and Norman gene pairs are encoded by aggregating the perturbation embeddings of their constituent genes. Targets absent from the expression gene set receive additional perturbation-only tokens without adding expression outputs. For Replogle, we apply the specified validation and test target lists within HepG2.

scBIG. scBIG organizes genes into modules and models responses with a module-aware encoder and conditional Flow Matching. We construct AnnData partitions using the benchmark splits and map conditions to its perturbation-slot format. Gene symbols are aligned with the GeneCompass and STRING resources used for gene-relation clustering. The resulting gene order is applied to all partitions, and predictions are restored to benchmark gene order for evaluation. We retain the ESM2 perturbation-feature interface. For Replogle, cell-type annotations are supplied as cell-line conditions, and validation and test targets are held out within HepG2. Data from the other cell lines remain eligible for training.

PerturbDiff. PerturbDiff uses functional diffusion. We train its public implementation from scratch on the benchmark gene expression data, retaining its Cross-DiT backbone and cell-set input interface. Dataset-specific configurations provide expression files, control labels, cell-type annotations, and held-out perturbations. For Adamson and Norman, the perturbation encoder identifies each perturbation's target genes and averages their learned embeddings. A Norman gene pair is therefore represented by its constituent genes rather than a separate combination identifier. Replogle uses the cell-type-conditioned holdout configuration. Predictions are evaluated in the benchmark expression space.

A.4 ADDITIONAL RESULTS

Table 4: Performance comparison on held-out perturbations in Norman. The best result is shown in **bold**, and the second-best is underlined.

Type	Method	RMSE All ↓	E-dist. All ↓	RMSE DE40 ↓	E-dist. DE40 ↓	DirAgr↑	LFCSpear↑	PDCorr↑	PDS_{cos}↑
Gra.	GEARS (Roohani et al., 2024)	0.058	0.768	0.186	0.810	0.772	0.653	0.461	0.531
Gra.	GraphVCI (Wu et al., 2022)	0.064	2.445	0.208	1.337	0.666	0.380	0.394	0.520
Fou.	scGPT (Cui et al., 2024)	0.048	2.033	0.142	1.094	0.850	0.786	0.584	0.803
Fou.	scFoundation (Hao et al., 2024)	0.095	1.998	0.117	1.074	0.715	0.581	0.404	0.570
Gen.	Departures (Chi et al., 2026a)	0.062	0.737	0.197	0.787	0.663	0.556	0.293	0.597
Gen.	Doloris (Chi et al., 2026b)	0.060	0.700	0.179	0.723	0.596	−0.047	0.398	0.672
Gen.	scDFM (Yu et al., 2026)	0.088	1.267	0.181	0.775	0.784	0.667	0.523	0.638
Gen.	scBIG (Ruan et al., 2026)	0.050	1.492	0.153	0.793	0.863	0.782	0.630	0.759
Gen.	PerturbDiff (Yuan et al., 2026)	0.047	0.665	0.138	0.611	0.747	0.239	0.669	0.786
	CoShare (Ours)	**0.038**	**0.400**	**0.089**	**0.375**	**0.911**	**0.857**	**0.747**	**0.851**