REWIRE THEN EXPRESS: CASCADED FLOW MATCHING FOR SINGLE-CELL PERTURBATION RESPONSE PREDICTION

Anonymous authors
Paper under double-blind review

ABSTRACT

Predicting cellular responses to genetic and chemical perturbations is a central challenge in functional genomics and drug discovery. Many recent models formulate this task as a single-stage transport from unperturbed to perturbed states. Inspired by the role of upstream regulatory changes in shaping transcriptional responses, we introduce *Rewire-then-Express*, a generative paradigm that structures perturbation prediction as an ordered cascade. The model first predicts an intermediate response representation, supervised by perturbation-induced shifts in pretrained single-cell embeddings, and then generates transcriptomic profiles conditioned on this representation. We instantiate this paradigm as **scCasFlow**, a **Cas**caded **Flow** Matching framework. Dual temporal coordinates separately parameterize intermediate-response and transcriptomic generation, while an ordered scheduling strategy coordinates their progression according to the intermediate-to-expression ordering. Flow matching operates in latent space, with a lightweight decoder enabling prediction across the retained gene vocabulary without highly variable gene selection. On the Norman and ComboSciPlex benchmarks, scCasFlow improves predictive performance over the evaluated methods under held-out perturbation splits and achieves more than 100× inference throughput of the evaluated expression-space flow matching baseline. Post hoc analyses further identify associations between the intermediate representations and transcription-factor activity scores inferred from held-out expression profiles, alongside agreement between predicted and observed pathway-aggregated expression changes. Together, these results support the predictive value of an explicitly supervised, ordered generative factorization for cellular perturbation modeling. Code is available at https://anonymous.4open.science/r/scCasFlow-80E8.

1 INTRODUCTION

Understanding how cells respond to genetic and chemical perturbations is a central challenge in functional genomics and drug discovery (Subramanian et al., 2017; Corsello et al., 2017). High-throughput single-cell omics, particularly single-cell RNA sequencing (scRNA-seq), have enabled these responses to be profiled at an unprecedented scale (Replogle et al., 2022; Srivatsan et al., 2020). However, the space of possible perturbations grows combinatorially with the number of genes and drugs, making exhaustive experimental screening impractical (Norman et al., 2019; Roohani et al., 2024). Predictive *in silico* models that generalize to unseen perturbations and their combinations are therefore essential for investigating regulatory interactions and prioritizing therapeutic combinations (Bunne et al., 2024).

Computational approaches to this problem range from latent-variable models that encode perturbation effects in learned representations (Lotfollahi et al., 2019; 2023) to diffusion- and flow-based models that generate perturbation-conditioned cell-state distributions (Luo et al., 2024; Chi et al., 2026b;a; Klein et al., 2025; Yu et al., 2026). Among these, Flow Matching (FM) (Lipman et al., 2023) has emerged as a flexible framework for learning continuous transformations of cellular-state distributions under perturbation (Klein et al., 2025; Yu et al., 2026). These flow-based approaches typically organize prediction as a single-stage conditional transition to the final transcriptomic response.

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.

Perturbation-induced change is thus learned within the same generative process that produces its expression outcome.

The dependence of gene expression on regulatory activity motivates an alternative organization of prediction. Perturbations can alter signaling and transcription-factor activity, which in turn shape downstream transcriptional responses (Purvis & Lahav, 2013; Hao & O'shea, 2012). This relationship suggests an ordered modeling strategy: first predict the change induced by a perturbation, and then generate expression conditioned on that prediction (Figure 1). We hypothesize that explicitly supervising perturbation-induced change as an intermediate in this process provides a useful inductive bias for response prediction.

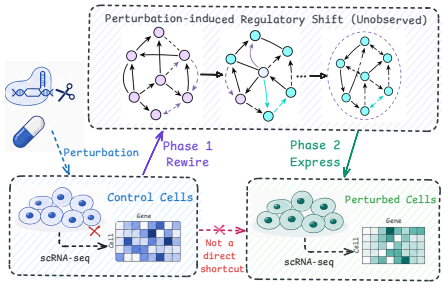

Figure 1: **Biological motivation for Rewire-then-Express.** Perturbation-induced changes in regulatory activity can shape downstream gene expression. This relationship motivates an ordered generative process that first predicts an intermediate response representation and then uses it to guide transcriptomic generation.

To this end, we introduce *Rewire-then-Express*, a generative paradigm in which perturbation-induced change serves as both a supervised prediction target and a condition for transcriptomic generation. The first stage predicts an intermediate response representation from the control-cell state and perturbation condition. Supervision is provided by shifts between control and perturbed cell populations in the embedding space of a pretrained single-cell foundation model (Dong et al., 2026). The second stage generates perturbed transcriptomic profiles conditioned on this intermediate representation, together with the control state and perturbation condition. In this way, the paradigm links population-level response supervision to the generation of individual cellular responses.

We instantiate this paradigm as **scCasFlow**, a **Cas**caded **Flow** Matching framework. Dual temporal coordinates separately parameterize the intermediate-response and expression flows, while an ordered scheduling strategy coordinates their progression. During training, expression-flow learning is conditioned on intermediate states near their supervised endpoints. At inference, the model completes intermediate-response generation before integrating the expression flow. The two flows share a neural vector field and operate in latent space, with a jointly optimized lightweight decoder mapping the generated expression representation to the retained gene vocabulary. This formulation reduces the dimensionality of iterative generation while enabling expression prediction without highly variable gene selection.

We evaluate scCasFlow on the Norman genetic-perturbation and ComboSciPlex chemical-perturbation benchmarks (Norman et al., 2019; Lotfollahi et al., 2023), testing generalization to held-out genes and unseen perturbation combinations. Across these settings, scCasFlow improves predictive performance over the evaluated methods and achieves more than 100× the inference throughput of the evaluated expression-space FM baseline, scDFM (Yu et al., 2026). Architectural ablations distinguish the benefits of the cascaded formulation from pretrained tokenizer reuse, while intermediate-target controls and matched training–inference schedule comparisons further examine the contributions of intermediate supervision and generation order. Further analyses reveal associations between predicted intermediate representations and transcription-factor activity scores inferred from held-out expression profiles, as well as agreement between predicted and observed pathway-aggregated expression changes. Together, these results support the predictive value of explicitly modeling perturbation-induced change as an intermediate that guides transcriptomic generation.

2 Related Work

Single-Cell Foundation Models. Single-cell foundation models (scFMs), including scBERT (Yang et al., 2022), scGPT (Cui et al., 2024), Geneformer (Theodoris et al., 2023), and scFoundation (Hao et al., 2024), learn representations of cellular states and gene-expression dependencies through large-scale pretraining. Subsequent developments incorporate spatial context with CellPLM (Wen et al., 2024), scale to 100 million cells with CellFM (Zeng et al., 2025), and enable zero-shot in-context learning with Stack (Dong et al., 2026). Nevertheless, recent benchmarks (Hou et al., 2026; Wei et al., 2026; Ahlmann-Eltze et al., 2025) show that improvements over classical baselines remain task-

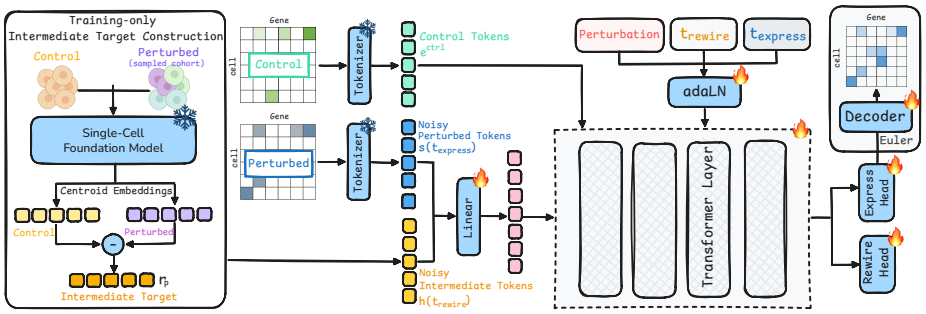

Figure 2: **Overview of scCasFlow.** During training, population shifts in frozen single-cell embeddings supervise an intermediate response, and both flows are jointly optimized under a phase-dependent schedule. At inference, the model first predicts the intermediate from control-cell information and the perturbation condition, then holds it fixed while generating the expression latent. A lightweight decoder maps the terminal latent to expression across the retained gene vocabulary.

dependent, with perturbation prediction presenting a persistent challenge. In scCasFlow, pretrained representations provide a space for defining perturbation-induced population shifts, which supervise the intermediate stage of the generative cascade.

Single-Cell Perturbation Response Prediction. Methods such as CPA (Lotfollahi et al., 2023), GEARS (Roohani et al., 2024), and PerturbNet (Yu et al., 2025) use latent representations or prior biological knowledge to generalize across perturbation conditions and combinations. Diffusion- and flow-based approaches, including scDiffusion (Luo et al., 2024), Doloris (Chi et al., 2026b), CellFlow (Klein et al., 2025), and scDFM (Yu et al., 2026), further formulate response prediction as conditional generation of perturbed cell distributions. These approaches predominantly model the response within a single-stage formulation. scCasFlow instead separates intermediate-response prediction from conditional expression generation, linking the two stages through explicit intermediate supervision and an ordered temporal schedule.

Temporal Scheduling in Generative Models. Time scheduling controls the progression of denoising in generative models, with conventional schedules designed around signal-to-noise allocation (Karras et al., 2022; Hoogeboom et al., 2023). Multi-schedule approaches, such as Diffusion Forcing (Chen et al., 2024) and Latent Forcing (Baade et al., 2026), allow different generative components to progress at different rates. scCasFlow adapts this principle to two latent flows, coordinating intermediate-response prediction and expression generation through an ordered temporal schedule. The resulting generation order serves as an inductive bias motivated by the role of upstream regulatory changes in shaping transcriptional responses (Purvis & Lahav, 2013).

3 Methodology

3.1 Preliminaries

Single-Cell Perturbation Response Prediction. Let $\mathcal{G}$ denote the gene vocabulary retained after preprocessing and alignment with the pretrained model, with $G = |\mathcal{G}|$. We predict expression across this vocabulary without further highly variable gene (HVG) selection. Let $\mathbf{x}^{\mathrm{ctrl}}, \mathbf{x}^{\mathrm{pert}} \in \mathbb{R}_{+}^{G}$ denote preprocessed control and perturbed expression profiles, respectively. A perturbation condition is represented by a multi-hot vector $\mathbf{p} \in \{0,1\}^{P}$, where P is the number of constituent genetic or chemical interventions in the perturbation library. Training tuples $(\mathbf{x}^{\mathrm{ctrl}}, \mathbf{p}, \mathbf{x}^{\mathrm{pert}})$ are formed by independently sampling a control cell from the control population and a perturbed cell from the population receiving $\mathbf{p}$. Our goal is to learn the conditional distribution

$$p_{\theta,\phi}(\mathbf{x}^{\mathrm{pert}} \mid \mathbf{x}^{\mathrm{ctrl}}, \mathbf{p}), \tag{1}$$

where θ parameterizes the neural vector field and its conditioning modules, and ϕ parameterizes the expression decoder.

Conditional Flow Matching. Conditional flow matching (CFM) (Lipman et al., 2023) learns a vector field that transports a base distribution to a condition-dependent target distribution. Let $\mathbf{c}$ denote the conditioning information and $q_1(\cdot \mid \mathbf{c})$ the corresponding target distribution. Given a base sample

$\mathbf{z}_0 \sim \mathcal{N}(\mathbf{0}, \mathbf{I}_d)$ and a target sample $\mathbf{z}_1 \sim q_1(\cdot \mid \mathbf{c})$, both in $\mathbb{R}^d$, we construct a linear interpolation path:

$$\mathbf{z}_t = (1-t)\mathbf{z}_0 + t\mathbf{z}_1, \qquad t \sim \mathcal{U}(0,1). \tag{2}$$

Here, d is the state dimension, $\mathbf{I}_d$ is the d-dimensional identity matrix, and $\mathcal{U}(0,1)$ denotes the uniform distribution over the generative time coordinate. The pathwise velocity is $\mathbf{z}_1 - \mathbf{z}_0$. A neural vector field $\mathcal{V}_\theta$ learns to predict this velocity by minimizing

$$\mathcal{L}_{\mathrm{CFM}}(\theta) = \mathbb{E}_{\mathbf{c},\mathbf{z}_0,\mathbf{z}_1,t}\left[\left\|\mathcal{V}_\theta(\mathbf{z}_t \mid t, \mathbf{c}) - (\mathbf{z}_1 - \mathbf{z}_0)\right\|_2^2\right], \tag{3}$$

where $\|\cdot\|_2$ denotes the Euclidean norm and the expectation is taken over the conditioning information, endpoint samples, and interpolation time. At inference, samples are generated by integrating the learned vector field from $t = 0$ to $t = 1$. scCasFlow extends this construction to two coordinated latent flows: intermediate-response prediction followed by conditional expression generation.

3.2 Rewire-then-Express: Ordered Generative Factorization

Rewire-then-Express couples shift prediction with conditional expression generation through an intermediate response representation $\mathbf{r} \in \mathbb{R}^D$, where D is the intermediate dimension. We first predict this representation from the control-cell state and perturbation condition, and then generate expression conditioned on it:

$$p_{\theta,\phi}(\mathbf{x}^{\mathrm{pert}} \mid \mathbf{x}^{\mathrm{ctrl}}, \mathbf{p}) = \int p_{\theta,\phi}(\mathbf{x}^{\mathrm{pert}} \mid \mathbf{x}^{\mathrm{ctrl}}, \mathbf{r}, \mathbf{p})\, p_\theta(\mathbf{r} \mid \mathbf{x}^{\mathrm{ctrl}}, \mathbf{p})\, \mathrm{d}\mathbf{r}. \tag{4}$$

The factor $p_\theta(\mathbf{r} \mid \mathbf{x}^{\mathrm{ctrl}}, \mathbf{p})$ models the intermediate response, while $p_{\theta,\phi}(\mathbf{x}^{\mathrm{pert}} \mid \mathbf{x}^{\mathrm{ctrl}}, \mathbf{r}, \mathbf{p})$ models expression conditioned on that response. We supervise this intermediate with population-level embedding shifts, linking perturbation-level response information to the generation of individual cellular profiles.

Intermediate Response Supervision. We construct intermediate targets using the frozen pretrained Stack encoder $\mathcal{E} : \mathbb{R}_+^G \to \mathbb{R}^D$ (Dong et al., 2026). Let $\mathcal{D}_{\mathrm{ctrl}}$ denote the training control population and $\mathcal{D}_{\mathbf{p}}$ the training population receiving perturbation $\mathbf{p}$. For each occurrence of $\mathbf{p}$ during training, we sample a cohort $\mathcal{S}_{\mathbf{p}}^K \subset \mathcal{D}_{\mathbf{p}}$ of K cells without replacement. Its embedding displacement relative to the control population is

$$\boldsymbol{\Delta}_{\mathbf{p}} = \frac{1}{K}\sum_{\mathbf{x}\in\mathcal{S}_{\mathbf{p}}^K} \mathcal{E}(\mathbf{x}) - \frac{1}{|\mathcal{D}_{\mathrm{ctrl}}|}\sum_{\mathbf{x}\in\mathcal{D}_{\mathrm{ctrl}}} \mathcal{E}(\mathbf{x}), \tag{5}$$

where $K = 128$ and $|\mathcal{D}_{\mathrm{ctrl}}|$ is the number of training control cells. We standardize this displacement feature-wise to obtain the intermediate target:

$$\mathbf{r}_{\mathbf{p}} = (\boldsymbol{\Delta}_{\mathbf{p}} - \boldsymbol{\mu}_{\boldsymbol{\Delta}}) \oslash \boldsymbol{\sigma}_{\boldsymbol{\Delta}}, \tag{6}$$

where $\boldsymbol{\mu}_{\boldsymbol{\Delta}}, \boldsymbol{\sigma}_{\boldsymbol{\Delta}} \in \mathbb{R}^D$ are the empirical feature-wise mean and standard deviation estimated from training perturbations, and $\oslash$ denotes element-wise division. The cohort is resampled during training, providing multiple population-shift estimates for each perturbation. Each sampled target augments the corresponding training tuple to $(\mathbf{x}^{\mathrm{ctrl}}, \mathbf{p}, \mathbf{r}_{\mathbf{p}}, \mathbf{x}^{\mathrm{pert}})$.

3.3 Cascaded Flow Matching with Ordered Temporal Scheduling

Dual Temporal Coordinates. We instantiate the ordered factorization with two latent flows that share a neural vector field. Separate temporal coordinates $t_{\mathrm{rewire}}, t_{\mathrm{express}} \in [0,1]$ parameterize progress in intermediate-response prediction (*Rewire*) and expression generation (*Express*), respectively. To perform expression generation in latent space, a frozen Stack tokenizer $\mathcal{T}$ maps the expression profiles to

$$\mathbf{e}^{\mathrm{ctrl}} = \mathcal{T}(\mathbf{x}^{\mathrm{ctrl}}), \qquad \mathbf{e}^{\mathrm{pert}} = \mathcal{T}(\mathbf{x}^{\mathrm{pert}}). \tag{7}$$

We write these latent states as vectors in $\mathbb{R}^{d_e}$, where d_e is their total dimension after flattening the token and feature axes for notation. The tokenizer $\mathcal{T}$ supplies the expression-flow states, whereas the full encoder $\mathcal{E}$ supplies the population embeddings used for intermediate supervision.

Starting from independent Gaussian samples $\mathbf{z}_{\text{rewire}} \sim \mathcal{N}(\mathbf{0}, \mathbf{I}_D)$ and $\mathbf{z}_{\text{express}} \sim \mathcal{N}(\mathbf{0}, \mathbf{I}_{d_e})$, we define the intermediate and expression paths as

$$\mathbf{h}(t_{\text{rewire}}) = (1 - t_{\text{rewire}})\mathbf{z}_{\text{rewire}} + t_{\text{rewire}}\mathbf{r}_{\mathbf{p}}, \tag{8}$$

$$\mathbf{s}(t_{\text{express}}) = (1 - t_{\text{express}})\mathbf{z}_{\text{express}} + t_{\text{express}}\mathbf{e}^{\text{pert}}, \tag{9}$$

where $\mathbf{h}(t_{\text{rewire}}) \in \mathbb{R}^D$ and $\mathbf{s}(t_{\text{express}}) \in \mathbb{R}^{d_e}$ are the interpolated intermediate and expression states. The shared vector field predicts both velocities:

$$[\hat{\mathbf{v}}_{\text{rewire}}, \hat{\mathbf{v}}_{\text{express}}] = \mathcal{V}_\theta\big(\mathbf{h}(t_{\text{rewire}}), \mathbf{s}(t_{\text{express}}) \mid t_{\text{rewire}}, t_{\text{express}}, \mathbf{e}^{\text{ctrl}}, \mathbf{p}\big). \tag{10}$$

Here, $\hat{\mathbf{v}}_{\text{rewire}} \in \mathbb{R}^D$ and $\hat{\mathbf{v}}_{\text{express}} \in \mathbb{R}^{d_e}$ estimate the pathwise velocity targets $\mathbf{r}_{\mathbf{p}} - \mathbf{z}_{\text{rewire}}$ and $\mathbf{e}^{\text{pert}} - \mathbf{z}_{\text{express}}$, respectively.

Ordered Temporal Scheduling. The two temporal coordinates allow the intermediate and expression flows to progress at different rates. Following the generation-ordering principle of latent forcing (Baade et al., 2026), we coordinate them through a phase-dependent schedule. Specifically, we sample a phase indicator $b \sim \text{Bernoulli}(\gamma)$, where $\gamma \in (0, 1)$ is the probability of selecting the Rewire phase, and draw the temporal coordinates as

$$\begin{cases} t_{\text{rewire}} \sim \mathcal{U}(0, 1), & t_{\text{express}} = 0, & b = 1, \\ t_{\text{rewire}} \sim \mathcal{U}(1 - \epsilon, 1), & t_{\text{express}} \sim \mathcal{U}(0, 1), & b = 0, \end{cases} \tag{11}$$

where ϵ controls the width of the intermediate flow's terminal region. In the Rewire phase ($b = 1$), the model learns intermediate-response generation while the expression state remains at its initial noise, $\mathbf{s}(0) = \mathbf{z}_{\text{express}}$. In the Express phase ($b = 0$), it learns expression generation conditioned on an intermediate state near its supervised endpoint. The terminal margin provides noise-augmented conditioning around the intermediate target.

Phase-Specific Flow Matching. The intermediate and expression velocity losses are

$$\ell_{\text{rewire}} = \|\hat{\mathbf{v}}_{\text{rewire}} - (\mathbf{r}_{\mathbf{p}} - \mathbf{z}_{\text{rewire}})\|_2^2, \quad \ell_{\text{express}} = \big\|\hat{\mathbf{v}}_{\text{express}} - (\mathbf{e}^{\text{pert}} - \mathbf{z}_{\text{express}})\big\|_2^2. \tag{12}$$

Under the ordered schedule, the cascaded objective activates the velocity loss associated with the sampled phase:

$$\mathcal{L}_{\text{Cascaded}}(\theta) = \mathbb{E}[b\,\ell_{\text{rewire}} + (1 - b)\,\ell_{\text{express}}]. \tag{13}$$

The expectation is taken over the sampled training tuples $(\mathbf{e}^{\text{ctrl}}, \mathbf{p}, \mathbf{r}_{\mathbf{p}}, \mathbf{e}^{\text{pert}})$, independent Gaussian noises $\mathbf{z}_{\text{rewire}}$ and $\mathbf{z}_{\text{express}}$, phase indicator b, and temporal coordinates drawn conditional on b according to Eq. (11). Both phases update the shared network parameters under this objective.

3.4 Latent-Space Architecture and Overall Optimization

Latent-Space Architecture. As illustrated in Figure 2, the shared vector field $\mathcal{V}_\theta$ uses Transformer layers based on scDFM (Yu et al., 2026). The intermediate and expression states are linearly projected and concatenated into a joint token stream. The network is conditioned on the control-cell latent $\mathbf{e}^{\text{ctrl}}$, perturbation condition $\mathbf{p}$, and temporal coordinates $(t_{\text{rewire}}, t_{\text{express}})$. Time and perturbation embeddings are incorporated through adaptive layer normalization (Peebles & Xie, 2023), and two output heads predict the intermediate and expression velocities. A lightweight decoder $\mathcal{D}_\phi : \mathbb{R}^{d_e} \to \mathbb{R}^G$ maps the terminal expression latent to the retained gene vocabulary. This architecture confines iterative generation to latent space while retaining expression-level supervision across all G genes.

For genetic perturbations, the perturbation encoder maps pretrained gene-identity representations from the fixed Stack vocabulary through a trainable projection shared across genes. The Stack tokenizer $\mathcal{T}$ and full encoder $\mathcal{E}$ remain frozen throughout training.

Expression Reconstruction and Overall Objective. To jointly optimize the expression flow and decoder without integrating an ordinary differential equation (ODE) at every training iteration, we estimate the terminal expression latent using a one-step Euler update:

$$\hat{\mathbf{e}}^{\text{pert}} = \mathbf{s}(t_{\text{express}}) + (1 - t_{\text{express}})\hat{\mathbf{v}}_{\text{express}}, \tag{14}$$

where $\hat{\mathbf{e}}^{\text{pert}}$ estimates the expression state at $t_{\text{express}} = 1$ from the current interpolated state and predicted velocity. Decoding this estimate provides expression-level supervision during the Express phase:

$$\mathcal{L}_{\text{Recon}}(\theta, \phi) = \mathbb{E}\left[(1-b)\left\|\mathcal{D}_\phi(\hat{\mathbf{e}}^{\text{pert}}) - \mathbf{x}^{\text{pert}}\right\|_2^2\right], \tag{15}$$

where the expectation follows the same training-sampling distribution as $\mathcal{L}_{\text{Cascaded}}$. The overall objective is

$$\mathcal{L}_{\text{Total}}(\theta, \phi) = \mathcal{L}_{\text{Cascaded}}(\theta) + \lambda \mathcal{L}_{\text{Recon}}(\theta, \phi), \tag{16}$$

where λ weights expression reconstruction. Thus, the Rewire phase optimizes intermediate-velocity prediction, whereas the Express phase jointly optimizes expression-velocity prediction and reconstruction.

4 Experiment

4.1 Experimental Setup

Datasets. We evaluate on two public single-cell perturbation benchmarks: Norman (Norman et al., 2019) (genetic perturbations) and ComboSciPlex (Lotfollahi et al., 2023) (chemical perturbations). We consider three standard generalization splits: *Norman-Additive* (unseen combinations of seen genes), *Norman-Holdout* (unseen genes), and *ComboSciPlex* (unseen drug combinations). Note that neither benchmark is included in the Stack/scGPT pre-training corpus, ensuring that our evaluations are conducted in a rigorous OOD setting without data exposure.

Baselines. We compare scCasFlow against four paradigms: **(i)** Control; **(ii)** Conditional predictive models (CPA (Lotfollahi et al., 2023), GEARS (Roohani et al., 2024), STATE (Adduri et al., 2025)); **(iii)** scFMs (scGPT (Cui et al., 2024), Stack (Dong et al., 2026)); and **(iv)** Flow matching models (CellFlow (Klein et al., 2025), scDFM (Yu et al., 2026)). For Stack, we employ its cellular representation foundation model rather than the in-context perturbation predictor. Because its pre-training strictly excludes perturbation data, using Stack as both baseline and encoder preserves the OOD integrity of perturbation-response tasks. We evaluate condition-level mean expression accuracy, perturbation-effect correlations, perturbation discrimination, and differential-expression signature agreement. Metric definitions are provided in Appendix.

Table 1: Comparison of different methods across evaluation metrics on the *Norman-Additive split.* **Bold** indicates the best performance, and underline indicates the second best.

Model	MSE↓	MAE↓	DS↑	Pearson Δ↑	Pearson Δ_{20}↑	Pearson Δ_{50}↑	DE Spearman LFC↑	DE Dir. Match↑	Overlap@50↑
Control	0.002823	0.019135	0.5161	–	-0.1640	-0.0601	-0.0035	0.3034	0.0000
CPA (2023) (Lotfollahi et al., 2023)	0.001962	0.018007	0.6171	0.4881	0.6804	0.7003	0.4065	0.7703	0.0097
GEARS (2024) (Roohani et al., 2024)	0.001759	0.018299	0.6410	0.5347	0.7293	0.7386	0.3695	0.7215	0.0052
scGPT (2024) (Cui et al., 2024)	0.002193	0.018607	0.5036	0.4709	0.4354	0.4861	-0.0619	0.6601	0.0039
CellFlow (2025) (Klein et al., 2025)	0.001337	0.014148	0.7024	0.7297	0.8275	0.8297	0.7377	0.9061	0.0484
STATE (2025) (Adduri et al., 2025)	0.002305	0.019546	0.5172	0.4288	0.4973	0.5228	0.2068	0.6910	0.0342
scDFM (2026) (Yu et al., 2026)	0.001752	0.020797	0.8897	0.6216	0.8320	0.8406	0.5523	0.8021	0.0361
Stack (2026) (Dong et al., 2026)	0.001156	0.016984	0.9407	0.7480	0.9087	0.9144	0.6172	0.8432	0.0884
scCasFlow	**0.000683**	**0.011581**	**0.9646**	**0.8488**	**0.9296**	**0.9374**	**0.8192**	**0.9587**	**0.1284**

Table 2: Comparison of different methods across evaluation metrics on the *Norman-Holdout split.* **Bold** indicates the best performance, and underline indicates the second best.

Model	MSE ↓	MAE ↓	DS ↑	Pearson Δ ↑	Pearson Δ_{20} ↑	Pearson Δ_{50} ↑	DE Spearman LFC ↑	DE Dir. Match ↑	Overlap@50 ↑
Control	0.002225	0.016680	0.5083	–	-0.1351	-0.0446	-0.0008	0.3066	0.0000
CPA (2023) (Lotfollahi et al., 2023)	0.001728	0.017776	0.5833	0.4408	0.6078	0.6270	0.3960	0.7430	0.0160
GEARS (2024) (Roohani et al., 2024)	0.001623	0.017730	0.6614	0.5303	0.6823	0.6930	0.3567	0.7116	0.0077
scGPT (2024) (Cui et al., 2024)	0.001863	0.017134	0.5078	0.4263	0.3603	0.4368	-0.0571	0.6341	0.0030
CellFlow (2025) (Klein et al., 2025)	0.001470	**0.014021**	0.6192	0.5579	0.6308	0.6504	0.6440	0.8300	0.0503
STATE (2025) (Adduri et al., 2025)	0.003020	0.023467	0.5011	0.3674	0.3640	0.4011	0.1418	0.6686	0.0290
scDFM (2026) (Yu et al., 2026)	0.002239	0.022141	0.6606	0.4840	0.6613	0.6717	0.5032	0.7544	0.0400
Stack (2026) (Dong et al., 2026)	0.001661	0.018612	0.6678	0.4336	0.5741	0.5966	0.4620	0.7043	0.0543
scCasFlow	**0.001326**	0.014201	**0.7436**	**0.6054**	**0.7175**	**0.7078**	**0.6671**	**0.8462**	**0.0800**

4.2 Comparison with State-of-the-Art

Combinatorial Generalization (Norman-Additive Split). scCasFlow achieves the lowest MSE of 0.000683 on unseen combinations of previously observed gene perturbations (Table 1). Its DE Spearman LFC reaches 0.8192, compared with 0.7377 for CellFlow and 0.5523 for scDFM,

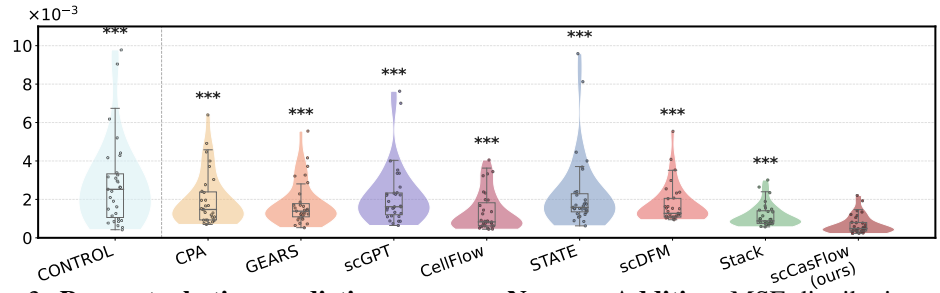

Figure 3: **Per-perturbation prediction errors on Norman-Additive.** MSE distributions across held-out perturbation combinations for each method. Paired Wilcoxon signed-rank tests compare each method with scCasFlow, with Benjamini–Hochberg false discovery rate correction across comparisons. Stars indicate adjusted significance levels: $^{*}p_{\text{adj}} < 0.05$, $^{**}p_{\text{adj}} < 0.01$, and $^{***}p_{\text{adj}} < 0.001$.

Table 3: **Prediction performance on ComboSciPlex.** Methods are evaluated on held-out drug combinations. **Bold** and underlining indicate the best and second-best performance, respectively.

Model	MSE ↓	MAE ↓	DS ↑	Pearson Δ ↑	Pearson Δ_{20} ↑	Pearson Δ_{50} ↑	DE Spearman LFC ↑	DE Dir. Match ↑	Overlap@50 ↑
CONTROL	0.021124	0.062302	0.6000	–	0.2140	0.0510	-0.0076	0.2820	0.0000
CPA (2023) (Lotfollahi et al., 2023)	0.019049	0.061522	0.6000	0.3364	0.7087	0.6101	0.3446	0.6274	0.0000
CellFlow (2025) (Klein et al., 2025)	0.006615	0.037420	0.8800	0.7819	0.9407	0.9169	0.8352	0.8982	0.4400
STATE (2025) (Adduri et al., 2025)	0.011554	0.054220	0.5200	0.6611	0.8367	0.8219	0.5242	0.7495	0.3200
scDFM (2026) (Yu et al., 2026)	0.027113	0.079908	0.8400	0.4629	0.9199	0.8881	0.7616	0.6090	0.2600
Stack (2026) (Dong et al., 2026)	0.004313	0.033507	0.8800	0.8383	0.9520	0.9502	0.8648	0.8685	0.4760
scCasFlow	**0.002821**	**0.024695**	**0.9600**	**0.8980**	**0.9542**	**0.9559**	**0.9374**	**0.9622**	**0.5080**

indicating closer agreement with observed differential-expression responses. It also outperforms the fine-tuned Stack model evaluated on the same split. The contribution of the cascaded formulation beyond the pretrained tokenizer is examined in the ablation study below. Figure 3 further summarizes the per-perturbation errors and paired statistical comparisons.

Generalization to Held-Out Genes (Norman-Holdout Split). When genes are held out as perturbation conditions during training, scCasFlow achieves a Pearson Δ of 0.6054 and an Overlap@50 of 0.0800 (Table 2). These exceed the corresponding scores of CellFlow (0.5579 and 0.0503) and Stack (0.4336 and 0.0543). The results extend the predictive advantage from new combinations of seen perturbations to combinations involving held-out genes.

Generalization to Chemical Perturbations (ComboSciPlex). scCasFlow achieves the best performance among the evaluated methods across all reported metrics on held-out drug combinations (Table 3). It obtains an MSE of 0.002821 and a Pearson Δ of 0.8980, compared with 0.006615 and 0.7819 for CellFlow, and 0.004313 and 0.8383 for Stack. Its DE Spearman LFC of 0.9374 and DE direction match of 0.9622 further indicate closer agreement in ranked expression changes and response directions. These results demonstrate the applicability of scCasFlow to both genetic and chemical perturbation prediction in the evaluated settings.

4.3 Ablation Study

Architectural Variants. We compare four variants to assess key design choices (Figure 4). *Monolithic Flow* removes the intermediate flow but retains the pretrained Stack tokenizer; its Overlap@50 drops by 84.4%, showing that tokenizer reuse alone cannot explain scCasFlow's performance. A randomly initialized compressor further confirms the benefit of pretrained latent representations. *Decoupled Decoder*, trained on observed-expression tokens rather than predicted terminal latents, performs worse, supporting training on model-generated states. *Deterministic Rewire*, which fixes $t_{\text{rewire}} = 1$ during expression-flow training, also degrades performance, favoring conditioning on near-terminal rather than only exact intermediate targets.

Perturbed Cohort Size. We vary K, the number of perturbed cells used to estimate the population-level intermediate target. Performance peaks at $K = 128$ among the tested values (Figure 4), with both smaller and larger cohorts yielding lower scores.

Intermediate Targets and Conditioning. We test whether intermediate supervision carries perturbation-specific information (Table 4a). Shuffling perturbation–target correspondences reduces Pearson Δ from 0.8488 to 0.7819, while random targets further reduce it to 0.7395; both controls

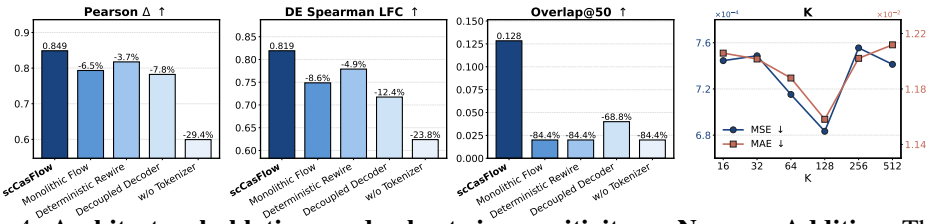

Figure 4: **Architectural ablations and cohort-size sensitivity on Norman-Additive.** The first three panels compare scCasFlow with four variants that modify the cascaded formulation, pretrained tokenizer, decoder optimization, or terminal-margin sampling. The final panel varies the number of perturbed cells used to construct the population-level intermediate target; $K = 128$ gives the strongest performance among the tested cohort sizes.

Table 4: **Intermediate-target controls and matched schedules on Norman-Additive.** (a) Intermediate-target and conditioning controls with the same backbone. (b) Dual-flow variants trained and evaluated with matched schedules and shared architecture/supervision; the expression-only reference removes the intermediate flow.

(a) Targets and conditioning

Variant	Pearson Δ ↑	DE LFC ↑
scCasFlow	**0.8488**	**0.8192**
Shuffled targets	0.7819	0.8102
Random targets	0.7395	0.7929
No control conditioning	-0.0436	0.3248
Zero perturbation embedding	0.5255	0.6422

(b) Matched training and inference schedules

Schedule	Setting	Pearson Δ ↑	DE LFC ↑
Rewire-then-Express	–	**0.8488**	**0.8192**
Intermediate-leading	$\alpha = 5$	0.8377	0.8148
Intermediate-leading	$\alpha = 3$	0.8327	0.8184
Simultaneous	$\alpha = 1$	0.8294	0.8158
Expression-leading	$\alpha = 0.33$	0.8159	0.8072
Expression-leading	$\alpha = 0.11$	0.8050	0.8069
Expression only (monolithic)	–	0.7938	0.7486

also decrease DE Spearman LFC. These results support the predictive value of the response targets and their alignment with perturbation conditions, beyond simply adding a latent prediction task. Removing control-cell conditioning lowers Pearson Δ to -0.0436, whereas zeroing the perturbation embedding lowers it to 0.5255, confirming contributions from both the control state and perturbation identity.

Generation Order under Matched Schedules. We test whether generating the intermediate response before expression provides a useful inductive bias by retraining the dual-flow model with matched training and inference schedules (Table 4b). Variants share the same architecture, supervision, and optimization, differing only in flow progression: intermediate-leading ($\alpha > 1$), simultaneous ($\alpha = 1$), or expression-leading ($\alpha < 1$); definitions are provided in Appendix A.2.3. Rewire-then-Express performs best, with Pearson $\Delta = 0.8488$ and DE Spearman LFC $= 0.8192$. Intermediate-leading alternatives achieve Pearson Δ of 0.8377 and 0.8327, versus 0.8294 for simultaneous and $0.8159/0.8050$ for expression-leading schedules. The expression-only reference reaches 0.7938 and 0.7486. Because all dual-flow variants retain the intermediate task, these results support ordered generation beyond simply adding a second flow and, together with the target controls, favor predicting supervised perturbation-induced change before expression.

4.4 Method Analysis

Associations with TF Activity. For each of the 31 unseen Norman-Additive combinations, we generate intermediate representations from 1,024 control cells and the perturbation condition, without accessing held-out perturbed expression. Activations are averaged across cells and token positions. After inference, transcription-factor (TF) activity shifts are estimated from held-out expression relative to controls using Univariate Linear Model scoring with CollecTRI (Müller-Dott et al., 2023; Badia-i Mompel et al., 2022). Neither the TF network nor these scores enters intermediate-target construction, training, or inference. All 16 channels exhibit significant Pearson correlations with TF activity shifts across perturbations after Benjamini–Hochberg correction over all 16×647 channel–TF pairs (Figure 5). As both intermediate supervision and TF readouts are expression-derived, these held-out associations support regulatory-associated information rather than direct recovery of upstream regulation.

Pathway-Level Response Agreement. We next examine whether the predicted transcriptomic responses preserve pathway-level structure. Figure 6 shows two representative pathways, H_2O_2 Catabolism and Oxygen Transport, selected according to observed response strength rather than

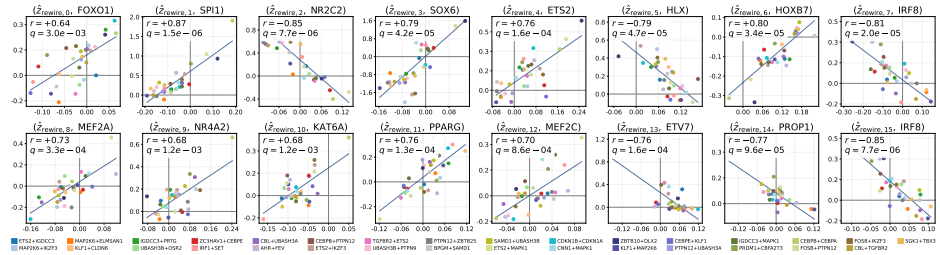

Figure 5: **Associations between predicted intermediate channels and inferred TF activity shifts.** Each panel shows a channel and its top-associated TF from the full channel–TF analysis, with one point per held-out perturbation combination.

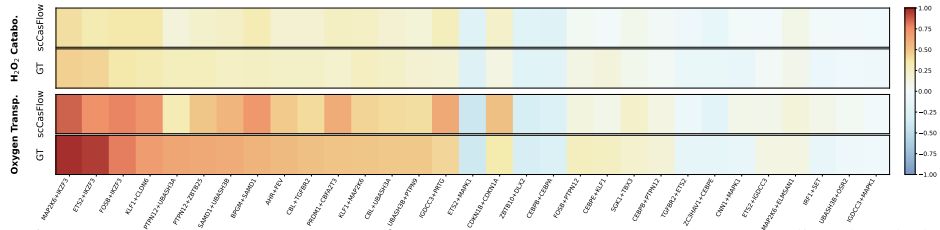

Figure 6: **Pathway-aggregated expression changes.** Heatmaps compare predicted and observed pathway-averaged expression changes relative to control for H_2O_2 Catabolism and Oxygen Transport across 31 held-out perturbation combinations. Both pathways were selected by observed response strength rather than prediction accuracy.

prediction accuracy. Across the 31 held-out Norman-Additive combinations, scCasFlow reproduces the perturbation-dependent response patterns of both pathways, with predicted pathway-level changes broadly following their observed counterparts. We then extend this analysis to 23 predefined K562-relevant pathways, comprising 14 GO and nine KEGG gene sets. For each pathway, we compute the Pearson correlation between predicted and observed pathway-averaged expression changes relative to control across the 31 held-out combinations. scCasFlow achieves a median correlation of 0.843, compared with 0.767 for scDFM, and performs better on 16 of 23 pathways (paired Wilcoxon test, $p = 0.004$). Because the two pathways shown in Figure 6 share a strong hemoglobin-related response, we further repeat the analysis after excluding hemoglobin genes from both predicted and observed profiles. For H_2O_2 Catabolism, the nine remaining genes yield correlations of 0.820 for scCasFlow and 0.747 for scDFM. The advantage also persists across the 20 hemoglobin-free pathways, indicating that the pathway-level agreement is not driven solely by the shared hemoglobin signal.

Computational Efficiency. We compare computational cost with the expression-space flow matching baseline scDFM on a single NVIDIA A800 80GB GPU (Table 5). Across the three evaluation settings, scCasFlow achieves 2.1–2.8× training-time speedups and an inference throughput of 26,712–29,448 cells per hour, corresponding to 164–194× that of scDFM under the evaluated configurations. Iterative generation is confined to a compact latent space, while a lightweight decoder maps the terminal expression latent to expression across the retained gene vocabulary without highly variable gene selection.

Table 5: **Computational efficiency.** Parentheses indicate training-time speedups or inference-throughput ratios relative to scDFM.

Metric / Method	Norman-Additive	Norman-Holdout	ComboSciPlex
Total training time (hours) ↓			
scDFM	38.1	38.1	29.3
scCasFlow	**17.9** (2.1×)	**16.3** (2.3×)	**10.5** (2.8×)
Inference throughput (cells / hour) ↑			
scDFM	138	144	180
scCasFlow	**26,712** (194×)	**27,072** (188×)	**29,448** (164×)

5 Conclusion

In this work, we introduce scCasFlow, a cascaded flow matching framework that instantiates Rewire-then-Express through population-level intermediate supervision and ordered latent-space generation. Across the evaluated genetic and chemical perturbation benchmarks, scCasFlow improves prediction for held-out perturbations and combinations. Intermediate-target controls and matched-schedule comparisons support the predictive value of explicitly supervising an intermediate response and generating it before expression.

AI USE STATEMENT

We used generative AI tools to assist with language editing, manuscript organization, and LaTeX formatting. We reviewed and verified all scientific claims, experimental results, and conclusions, and take full responsibility for the final content of this work.

REFERENCES

Abhinav K Adduri, Dhruv Gautam, Beatrice Bevilacqua, Alishba Imran, Rohan Shah, Mohsen Naghipourfar, Noam Teyssier, Rajesh Ilango, Sanjay Nagaraj, Mingze Dong, et al. Predicting cellular responses to perturbation across diverse contexts with state. *BioRxiv*, pp. 2025–06, 2025.

Constantin Ahlmann-Eltze, Wolfgang Huber, and Simon Anders. Deep-learning-based gene perturbation effect prediction does not yet outperform simple linear baselines. *Nature Methods*, 22(8): 1657–1661, 2025.

Alan Baade, Eric Ryan Chan, Kyle Sargent, Changan Chen, Justin Johnson, Ehsan Adeli, and Li Fei-Fei. Latent forcing: Reordering the diffusion trajectory for pixel-space image generation. *arXiv preprint arXiv:2602.11401*, 2026.

Pau Badia-i Mompel, Jesús Vélez Santiago, Jana Braunger, Celina Geiss, Daniel Dimitrov, Sophia Müller-Dott, Petr Taus, Aurelien Dugourd, Christian H Holland, Ricardo O Ramirez Flores, et al. decoupler: ensemble of computational methods to infer biological activities from omics data. *Bioinformatics advances*, 2, 2022.

Yoav Benjamini and Yosef Hochberg. Controlling the false discovery rate: a practical and powerful approach to multiple testing. *Journal of the Royal statistical society: series B (Methodological)*, 57(1):289–300, 1995.

Charlotte Bunne, Yusuf Roohani, Yanay Rosen, Ankit Gupta, Xikun Zhang, Marcel Roed, Theo Alexandrov, Mohammed AlQuraishi, Patricia Brennan, Daniel B Burkhardt, et al. How to build the virtual cell with artificial intelligence: Priorities and opportunities. *Cell*, 187(25):7045–7063, 2024.

Boyuan Chen, Diego Martí Monsó, Yilun Du, Max Simchowitz, Russ Tedrake, and Vincent Sitzmann. Diffusion forcing: Next-token prediction meets full-sequence diffusion. *Advances in Neural Information Processing Systems*, 37:24081–24125, 2024.

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, Zelin Zang, and Stan Z. Li. Doloris: Dual conditional diffusion implicit bridges with sparsity masking strategy for unpaired single-cell perturbation estimation. In *The Fourteenth International Conference on Learning Representations*, 2026b. URL https://openreview.net/forum?id=rvpDHfoTd2.

Steven M Corsello, Joshua A Bittker, Zihan Liu, Joshua Gould, Patrick McCarren, Jodi E Hirschman, Stephen E Johnston, Anita Vrcic, Bang Wong, Mariya Khan, et al. The drug repurposing hub: a next-generation drug library and information resource. *Nature medicine*, 23(4):405–408, 2017.

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.

Mingze Dong, Abhinav Adduri, Dhruv Gautam, Christopher Carpenter, Rohan Shah, Chiara Ricci-Tam, Yuval Kluger, Dave P Burke, and Yusuf H Roohani. Stack: In-context learning of single-cell biology. *bioRxiv*, pp. 2026–01, 2026.

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.

Nan Hao and Erin K O'shea. Signal-dependent dynamics of transcription factor translocation controls gene expression. *Nature structural & molecular biology*, 19(1):31–39, 2012.

Emiel Hoogeboom, Jonathan Heek, and Tim Salimans. simple diffusion: End-to-end diffusion for high resolution images. In *International Conference on Machine Learning*, pp. 13213–13232. PMLR, 2023.

Siyu Hou, Penghui Yang, Wenjing Ma, Jade Xiaoqing Wang, and Xiang Zhou. A unified framework enables accessible deployment and comprehensive benchmarking of single-cell foundation models. *bioRxiv*, pp. 2026–01, 2026.

Tero Karras, Miika Aittala, Timo Aila, and Samuli Laine. Elucidating the design space of diffusion-based generative models. *Advances in neural information processing systems*, 35:26565–26577, 2022.

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*, pp. 2025–04, 2025.

Yaron Lipman, Ricky T. Q. Chen, Heli Ben-Hamu, Maximilian Nickel, and Matthew Le. Flow matching for generative modeling. In *The Eleventh International Conference on Learning Representations*, 2023. URL https://openreview.net/forum?id=PqvMRDCJT9t.

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.

Erpai Luo, Minsheng Hao, Lei Wei, and Xuegong Zhang. scdiffusion: conditional generation of high-quality single-cell data using diffusion model. *Bioinformatics*, 40(9):btae518, 2024.

Lukas Mathur, B Szalai, NH Du, Ramesh Utharala, Martine Ballinger, JJM Landry, M Ryckelynck, Vladimir Benes, Julio Saez-Rodriguez, and Christoph A Merten. Combi-seq for multiplexed transcriptome-based profiling of drug combinations using deterministic barcoding in single-cell droplets. *Nature communications*, 13(1):4450, 2022.

Sophia Müller-Dott, Eirini Tsirvouli, Miguel Vazquez, Ricardo O Ramirez Flores, Pau Badia-i Mompel, Robin Fallegger, Dénes Türei, Astrid Lægreid, and Julio Saez-Rodriguez. Expanding the coverage of regulons from high-confidence prior knowledge for accurate estimation of transcription factor activities. *Nucleic acids research*, 51(20):10934–10949, 2023.

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.

William Peebles and Saining Xie. Scalable diffusion models with transformers. In *Proceedings of the IEEE/CVF international conference on computer vision*, pp. 4195–4205, 2023.

Stefan Peidli, Tessa D Green, Ciyue Shen, Torsten Gross, Joseph Min, Samuele Garda, Bo Yuan, Linus J Schumacher, Jake P Taylor-King, Debora S Marks, et al. scperturb: harmonized single-cell perturbation data. *Nature Methods*, 21(3):531–540, 2024.

Jeremy E Purvis and Galit Lahav. Encoding and decoding cellular information through signaling dynamics. *Cell*, 152(5):945–956, 2013.

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.

Sanjay R Srivatsan, José L McFaline-Figueroa, Vijay Ramani, Lauren Saunders, Junyue Cao, Jonathan Packer, Hannah A Pliner, Dana L Jackson, Riza M Daza, Lena Christiansen, et al. Massively multiplex chemical transcriptomics at single-cell resolution. *Science*, 367(6473):45–51, 2020.

Aravind Subramanian, Rajiv Narayan, Steven M Corsello, David D Peck, Ted E Natoli, Xiaodong Lu, Joshua Gould, John F Davis, Andrew A Tubelli, Jacob K Asiedu, et al. A next generation connectivity map: L1000 platform and the first 1,000,000 profiles. *Cell*, 171(6):1437–1452, 2017.

Christina V Theodoris, Ling Xiao, Anant Chopra, Mark D Chaffin, Zeina R Al Sayed, Matthew C Hill, Helene Mantineo, Elizabeth M Brydon, Zexian Zeng, X Shirley Liu, et al. Transfer learning enables predictions in network biology. *Nature*, 618(7965):616–624, 2023.

Zhiting Wei, Yiheng Wang, Yicheng Gao, Shuguang Wang, Ping Li, Duanmiao Si, Yuli Gao, Siqi Wu, Danlu Li, Kejing Dong, et al. Benchmarking algorithms for generalizable single-cell perturbation response prediction. *Nature Methods*, 23(2):451–464, 2026.

Hongzhi Wen, Wenzhuo Tang, Xinnan Dai, Jiayuan Ding, Wei Jin, Yuying Xie, and Jiliang Tang. CellPLM: Pre-training of cell language model beyond single cells. In *The Twelfth International Conference on Learning Representations*, 2024. URL https://openreview.net/forum?id=BKXvPDekud.

Fan Yang, Wenchuan Wang, Fang Wang, Yuan Fang, Duyu Tang, Junzhou Huang, Hui Lu, and Jianhua Yao. scbert as a large-scale pretrained deep language model for cell type annotation of single-cell rna-seq data. *Nature machine intelligence*, 4(10):852–866, 2022.

Nicholas D Youngblut, Christopher Carpenter, Jaanak Prashar, Chiara Ricci-Tam, Rajesh Ilango, Noam Teyssier, Silvana Konermann, Patrick D Hsu, Alexander Dobin, David P Burke, et al. scbasecount: an ai agent-curated, uniformly processed, and continually expanding single cell data repository. *bioRxiv*, pp. 2025–02, 2025.

Chenglei Yu, Chuanrui Wang, Bangyan Liao, and Tailin Wu. scDFM: Distributional flow matching model for robust single-cell perturbation prediction. In *The Fourteenth International Conference on Learning Representations*, 2026. URL https://openreview.net/forum?id=QSGanMEcUV.

Hengshi Yu, Weizhou Qian, Yuxuan Song, and Joshua D Welch. Perturbnet predicts single-cell responses to unseen chemical and genetic perturbations. *Molecular Systems Biology*, 21(8):960, 2025.

Yuansong Zeng, Jiancong Xie, Ningyuan Shangguan, Zhuoyi Wei, Wenbing Li, Yun Su, Shuangyu Yang, Chengyang Zhang, Jinbo Zhang, Nan Fang, et al. Cellfm: a large-scale foundation model pre-trained on transcriptomics of 100 million human cells. *Nature Communications*, 16(1):4679, 2025.

APPENDIX

A ADDITIONAL DETAILS

A.1 DATASET AND METRIC DETAILS

A.1.1 DATASETS

Norman dataset. The Norman dataset (Norman et al., 2019) is a CRISPR activation (CRISPRa) Perturb-seq benchmark collected in K562 cells. We used the harmonized AnnData version distributed by scPerturb (Peidli et al., 2024) to define two generalization settings on this dataset.

Expression values were normalized in the original scPerturb gene space using total-count normalization with target sum 10^4, followed by `log1p` transformation. The normalized matrix was aligned to the gene vocabulary of the pretrained Stack (Dong et al., 2026) encoder by retaining genes shared between the scPerturb annotation and the Stack vocabulary. To maintain a consistent input space across conditions, perturbation conditions involving genes outside this vocabulary were excluded. The processed Norman data contain 95,181 cells and 14,129 genes, comprising one control condition together with 88 single-gene and 105 double-gene perturbation conditions. No highly variable gene selection was applied, and all retained genes were used as prediction targets.

Two complementary train–test splits were used to assess different generalization regimes. The *Norman-Additive* split evaluates combinatorial interpolation: all single-gene perturbations are observed during training, while a subset of double-gene perturbations is reserved for testing. Thus, test conditions are unseen gene combinations whose individual components are available in the training set. The *Norman-Holdout* split evaluates unseen-gene generalization: selected perturbation genes are excluded from training, and all single- or double-gene perturbation conditions involving these genes are assigned to the test set. All splits were generated with a fixed random seed to ensure reproducible train–test assignment.

ComboSciPlex dataset. The ComboSciPlex dataset (Mathur et al., 2022) is a combinatorial drug-perturbation benchmark on A549 cells, extending the sci-Plex profiling strategy (Srivatsan et al., 2020) to paired drug treatments. Experiments used the released log-normalized expression matrix.

For consistency with the Norman setting and the pretrained encoder vocabulary, the expression space was restricted to genes shared between the ComboSciPlex gene annotation and the Stack vocabulary. The processed ComboSciPlex data contain 63,378 cells and 12,855 genes across 32 conditions, comprising one control condition together with six single-drug and 25 double-drug conditions. No highly variable gene selection was applied, and all retained genes were used as prediction targets.

For this benchmark, we followed the dataset-defined split. The reported *ComboSciPlex* split evaluates unseen double-drug combinations, corresponding to chemical combination generalization.

To account for the use of pretrained Stack representations, we examined the provenance of the evaluation benchmarks relative to the Stack pretraining corpus. The Stack checkpoint was pretrained on scBaseCount (Youngblut et al., 2025), an observational single-cell corpus without perturbation labels or perturbation-response supervision. Dataset-level matching using public accession identifiers and study-level metadata indicated that neither Norman nor ComboSciPlex was included in the Stack pretraining corpus. Thus, the downstream evaluations are not affected by direct benchmark-level overlap with the pretrained representation model.

A.1.2 METRICS CALCULATION

Notation and implementation. For each test perturbation condition $p \in \mathcal{P}_{\text{test}}$, let $\mathbf{X}_p^{\text{obs}} \in \mathbb{R}^{n_p^{\text{obs}} \times G}$ and $\mathbf{X}_p^{\text{pred}} \in \mathbb{R}^{n_p^{\text{pred}} \times G}$ denote the observed and predicted single-cell expression matrices over G evaluation genes. Their condition-level mean profiles are denoted by $\boldsymbol{\mu}_p^{\text{obs}}$ and $\boldsymbol{\mu}_p^{\text{pred}}$, respectively. We define the gene-expression perturbation effect as the mean shift from the corresponding control condition:

$$\boldsymbol{\delta}_p^* = \boldsymbol{\mu}_p^* - \boldsymbol{\mu}_{\text{ctrl}}^*, \qquad * \in \{\text{obs}, \text{pred}\}. \tag{17}$$

Unless otherwise specified, each metric is computed independently for each perturbation condition and then averaged over $\mathcal{P}_{\text{test}}$. MSE, MAE, Pearson Δ, Discrimination Score (DS), and the differential

expression (DE)-based metrics are computed using `cell-eval` (Adduri et al., 2025). Pearson Δ_{20} and Pearson Δ_{50} are computed separately as top-K perturbation-effect correlations.

Expression-level accuracy. MSE and MAE evaluate the discrepancy between predicted and observed condition-level mean expression profiles. For each perturbation condition, they are defined as

$$\mathrm{MSE}_p = \frac{1}{G}\sum_{g=1}^{G}\left(\mu_{p,g}^{\mathrm{pred}} - \mu_{p,g}^{\mathrm{obs}}\right)^2, \tag{18}$$

$$\mathrm{MAE}_p = \frac{1}{G}\sum_{g=1}^{G}\left|\mu_{p,g}^{\mathrm{pred}} - \mu_{p,g}^{\mathrm{obs}}\right|. \tag{19}$$

Both metrics quantify expression-level fidelity, with MSE placing greater emphasis on large gene-wise deviations and MAE measuring the average absolute discrepancy. Lower values indicate closer agreement with the observed mean expression profile.

Perturbation-effect consistency and discriminability. Pearson Δ evaluates whether the model recovers the global perturbation-induced expression change. For each condition, it computes the Pearson correlation between predicted and observed perturbation-effect vectors:

$$\text{Pearson } \Delta_p = \mathrm{corr}_{\mathrm{Pearson}}\left(\boldsymbol{\delta}_p^{\mathrm{pred}}, \boldsymbol{\delta}_p^{\mathrm{obs}}\right). \tag{20}$$

Higher values indicate stronger agreement in the direction and relative gene-wise magnitude of the perturbation response.

Discrimination Score (DS) measures perturbation-level discriminability. In our implementation, DS is computed using the L_2 distance between predicted and observed perturbation-effect vectors. For each predicted perturbation effect, its distance to every observed perturbation effect is computed as

$$d_{p,q} = \left\|\boldsymbol{\delta}_p^{\mathrm{pred}} - \boldsymbol{\delta}_q^{\mathrm{obs}}\right\|_2, \qquad q \in \mathcal{P}_{\mathrm{test}}. \tag{21}$$

Let r_p denote the zero-based rank of the matched observed condition among all candidate conditions sorted by $d_{p,q}$. The per-condition score is defined as

$$\mathrm{DS}_p = 1 - \frac{r_p}{|\mathcal{P}_{\mathrm{test}}|}. \tag{22}$$

A higher DS indicates that the predicted response is more readily matched to its corresponding observed perturbation, reflecting stronger condition-specific structure.

Top perturbed gene correlations. Pearson Δ_{20} and Pearson Δ_{50} focus on genes with the strongest observed perturbation responses. For each condition p, we select the top-K genes according to the absolute observed perturbation effect:

$$\mathcal{T}_p^{(K)} = \mathrm{TopK}\left(\left|\boldsymbol{\delta}_p^{\mathrm{obs}}\right|\right). \tag{23}$$

The top-K perturbation-effect correlation is then computed as

$$\text{Pearson } \Delta_{K,p} = \mathrm{corr}_{\mathrm{Pearson}}\left(\boldsymbol{\delta}_p^{\mathrm{pred}}[\mathcal{T}_p^{(K)}], \boldsymbol{\delta}_p^{\mathrm{obs}}[\mathcal{T}_p^{(K)}]\right), \qquad K \in \{20, 50\}. \tag{24}$$

We report this metric for $K = 20$ and $K = 50$ as Pearson Δ_{20} and Pearson Δ_{50}, respectively. These metrics complement the genome-wide Pearson Δ by emphasizing genes with the largest observed perturbation-induced changes.

Differential expression signature consistency. Differential expression is computed for each perturbation against the control condition using the `pdex` backend in `cell-eval`, with a Wilcoxon rank-sum test followed by Benjamini–Hochberg false discovery rate correction. The observed significant DE gene set is defined as

$$\mathcal{S}_p^{\mathrm{obs}} = \{g : \mathrm{FDR}_{p,g}^{\mathrm{obs}} < 0.05\}. \tag{25}$$

DE Spearman LFC measures the rank agreement between predicted and observed log-fold changes over the observed significant DE genes:

$$\text{DE Spearman LFC}_p = \mathrm{corr}_{\mathrm{Spearman}}\left(\{\mathrm{LFC}_{p,g}^{\mathrm{pred}}\}_{g\in\mathcal{S}_p^{\mathrm{obs}}}, \{\mathrm{LFC}_{p,g}^{\mathrm{obs}}\}_{g\in\mathcal{S}_p^{\mathrm{obs}}}\right). \tag{26}$$

Table S1: Summary of scCasFlow hyperparameters.

Symbol	Meaning	Value
K	Perturbed cohort size for intermediate target estimation	128
γ	Probability of selecting the Rewire phase	0.4
ϵ	Terminal margin for intermediate conditioning	0.25
λ	Reconstruction loss weight	0.1
B	Training batch size	64
–	Optimizer	AdamW
η_0	Initial learning rate	5×10^{-5}
E	Training epochs	150
N_{warm}	Learning-rate warmup steps	2000
N_{rewire}	Rewire integration steps at inference	20
N_{express}	Express integration steps at inference	20

DE Dir. Match evaluates directional consistency by computing the fraction of observed significant DE genes whose predicted and observed log-fold changes have the same sign:

$$\text{DE Dir. Match}_p = \frac{1}{|\mathcal{S}_p^{\text{obs}}|} \sum_{g \in \mathcal{S}_p^{\text{obs}}} \mathbb{I}\left[\text{sign}\left(\text{LFC}_{p,g}^{\text{pred}}\right) = \text{sign}\left(\text{LFC}_{p,g}^{\text{obs}}\right)\right]. \tag{27}$$

For Overlap@50, significant DE genes are first defined separately for the observed and predicted profiles:

$$\mathcal{S}_p^* = \{g : \text{FDR}_{p,g}^* < 0.05\}, \qquad * \in \{\text{obs}, \text{pred}\}. \tag{28}$$

We then rank significant DE genes by absolute log-fold change and define the top-50 DE gene sets as

$$\mathcal{G}_p^{*,50} = \arg\operatorname*{top}_{50} \left\{\left|\text{LFC}_{p,g}^*\right| : g \in \mathcal{S}_p^*\right\}, \qquad * \in \{\text{obs}, \text{pred}\}. \tag{29}$$

Overlap@50 is computed as

$$\text{Overlap@50}_p = \frac{|\mathcal{G}_p^{\text{obs},50} \cap \mathcal{G}_p^{\text{pred},50}|}{\min\left(50, |\mathcal{G}_p^{\text{obs},50}|\right)}. \tag{30}$$

Higher DE Spearman LFC, DE Dir. Match, and Overlap@50 values indicate stronger consistency with the observed DE signature in ranking, direction, and gene-set composition, respectively.

A.2 Implementation Details

A.2.1 Hyper-parameter Settings

To ensure the reproducibility of our experiments, all key hyperparameters used for the training and inference phases of our proposed methodology are summarized in Table S1. All experiments were run on a single NVIDIA A800 80GB GPU.

A.2.2 Details of the Association Analysis between Rewire-Phase Latent Channels and Inferred TF Activity Shifts

TF activity shift inference. To obtain a post hoc TF activity readout for each perturbation, we inferred transcription factor (TF) activity from held-out perturbed single-cell transcriptomic profiles and the control population using the Univariate Linear Model (ULM) implemented in `decoupler` v2.1.6 (Badia-i Mompel et al., 2022). The TF–target prior was provided by the CollecTRI human network, which contains 1,185 TFs and 42,990 signed TF–target edges, with positive and negative weights denoting activation and repression, respectively (Müller-Dott et al., 2023).

For each cell c and TF t, ULM estimates the association between the expression profile of the cell and the signed target profile of the TF:

$$y_{c,g} = \beta_{0,c,t} + \beta_{1,c,t} w_{g,t} + \varepsilon_{c,g}, \quad g = 1, \ldots, G, \tag{31}$$

where $y_{c,g}$ denotes the log-normalised expression of gene g in cell c, and $w_{g,t}$ is the signed regulatory weight from TF t to gene g. The t-statistic associated with the slope estimate $\hat{\beta}_{1,c,t}$ was used as the inferred TF activity score, denoted by $A_{c,t}$. TFs with fewer than five target genes present in the expression matrix were excluded, resulting in 647 TFs for downstream analysis.

For each perturbation condition p, the inferred TF activity shift was computed relative to the control population:

$$\Delta\mathrm{TFA}_{p,t} = \frac{1}{|\mathcal{C}_p|}\sum_{c\in\mathcal{C}_p} A_{c,t} - \frac{1}{|\mathcal{C}_{\mathrm{ctrl}}|}\sum_{c\in\mathcal{C}_{\mathrm{ctrl}}} A_{c,t}, \tag{32}$$

where $\mathcal{C}_p$ and $\mathcal{C}_{\mathrm{ctrl}}$ denote the cells from perturbation condition p and the control condition, respectively. This procedure yields a perturbation-level TF activity shift matrix

$$\Delta\mathrm{TFA} \in \mathbb{R}^{31\times 647}, \tag{33}$$

covering the 31 entirely unseen combinatorial perturbations used in the rewire-phase analysis.

Rewire-phase latent representation computation. For the same set of 31 unseen combinatorial perturbations, we generated the rewire-phase latent representation $\hat{z}_{\mathrm{rewire}}$ using scCasFlow conditioned on 1,024 control cells. The generated representation contains token-wise latent activations with 16 rewire channels. To obtain a perturbation-level summary for each channel, we averaged the generated activations over both sampled control cells and token positions:

$$\bar{z}_{\mathrm{rewire},p,k} = \frac{1}{NL}\sum_{i=1}^{N}\sum_{\ell=1}^{L}\hat{z}_{\mathrm{rewire},p,i,\ell,k}, \quad N = 1024,\ L = 100, \tag{34}$$

where p indexes perturbation conditions, i indexes sampled control cells, ℓ indexes token positions, and k indexes rewire channels. This gives a compact rewire-phase representation

$$\bar{z}_{\mathrm{rewire}} \in \mathbb{R}^{31\times 16}. \tag{35}$$

Because the intermediate target is defined from a standardized perturbation-induced displacement in pretrained embedding space during training, $\bar{z}_{\mathrm{rewire}}$ summarizes the generated intermediate response at the perturbation level rather than an absolute cellular state.

Associations between rewire channels and TF activity shifts. We quantified associations between the generated rewire-phase latent channels and inferred TF activity shifts by correlating each rewire channel with each TF activity shift across perturbation conditions. Specifically, for channel k and TF t, we computed the Pearson correlation coefficient across the 31 perturbations:

$$r_{k,t} = \mathrm{corr}\left(\bar{z}_{\mathrm{rewire},:,k}, \Delta\mathrm{TFA}_{:,t}\right). \tag{36}$$

Equivalently,

$$r_{k,t} = \frac{\sum_{p=1}^{31}\left(\bar{z}_{\mathrm{rewire},p,k} - \mu_{\bar{z},k}\right)\left(\Delta\mathrm{TFA}_{p,t} - \mu_{\mathrm{TFA},t}\right)}{\sqrt{\sum_{p=1}^{31}\left(\bar{z}_{\mathrm{rewire},p,k} - \mu_{\bar{z},k}\right)^2}\sqrt{\sum_{p=1}^{31}\left(\Delta\mathrm{TFA}_{p,t} - \mu_{\mathrm{TFA},t}\right)^2}}, \tag{37}$$

where $\mu_{\bar{z},k}$ and $\mu_{\mathrm{TFA},t}$ denote the mean values of rewire channel k and TF activity shift t across perturbation conditions, respectively.

A two-sided significance test was performed for each channel–TF pair under the null hypothesis $H_0 : \rho_{k,t} = 0$. With $n = 31$ perturbation conditions, the test statistic is

$$T_{k,t} = \frac{r_{k,t}\sqrt{n-2}}{\sqrt{1-r_{k,t}^2}}, \tag{38}$$

which follows a Student's t-distribution with $n - 2 = 29$ degrees of freedom under the null.

Multiple testing correction. Since all 16×647 channel–TF pairs were tested, we controlled the false discovery rate using the Benjamini–Hochberg procedure (Benjamini & Hochberg, 1995). Let

$$p_{(1)} \le p_{(2)} \le \cdots \le p_{(m)}, \quad m = 16 \times 647, \tag{39}$$

Table S2: Supplementary comparison of different methods across evaluation metrics on the Norman holdout split, including both single- and double-perturbation settings. **Bold** indicates the best performance, and underline indicates the second-best performance.

Setting	Model	MSE ↓	MAE ↓	DS ↑	Pearson Δ ↑	Pearson Δ_{20} ↑	Pearson Δ_{50} ↑	DE Spearman LFC ↑	DE Dir. Match ↑	Overlap@50 ↑
Single	Control	0.000951	0.010832	0.5417	–	-0.1128	-0.0425	0.0078	0.3024	0.0000
	CPA (2023) (Lotfollahi et al., 2023)	0.001050	0.015244	**0.6042**	0.3161	0.3908	0.4090	0.3059	0.6878	0.0200
	GEARS (2024) (Roohani et al., 2024)	0.000947	0.012949	0.5694	0.3589	0.4235	0.4208	0.2645	0.6554	0.0100
	scGPT (2024) (Cui et al., 2024)	0.001104	0.014369	0.5208	0.3403	0.2876	0.2918	-0.0367	0.6005	0.0050
	CellFlow (2025) (Klein et al., 2025)	**0.000835**	**0.010239**	0.5556	0.3567	**0.5344**	**0.4688**	0.4950	**0.7315**	0.0250
	STATE (2025) (Adduri et al., 2025)	0.002928	0.023742	0.5417	0.2728	0.2550	0.2696	0.0998	0.6252	0.0200
	scDFM (2026) (Yu et al., 2026)	0.001512	0.018692	0.5556	0.3018	0.4245	0.4223	0.4266	0.6924	0.0400
	Stack (2026) (Dong et al., 2026)	0.001195	0.016008	0.5417	0.1442	0.1708	0.2006	0.3444	0.6048	0.0200
	scCasFlow	0.000965	0.011671	0.5833	**0.3688**	0.5106	0.4285	**0.5242**	0.7246	**0.0467**
Double	Control	0.002544	0.018142	0.5104	–	-0.1406	-0.0451	-0.0026	0.3023	0.0000
	CPA (2023) (Lotfollahi et al., 2023)	0.001897	0.018409	0.5872	0.4720	0.6621	0.6815	0.3992	0.7566	0.0146
	GEARS (2024) (Roohani et al., 2024)	0.001792	0.018925	0.6762	0.5731	0.7471	0.7611	0.4024	0.7253	0.0071
	scGPT (2024) (Cui et al., 2024)	0.002053	0.017825	0.5091	0.4478	0.3785	0.4731	-0.0592	0.6429	0.0025
	CellFlow (2025) (Klein et al., 2025)	0.001628	0.014967	0.6411	0.6082	0.6549	0.6958	0.6507	0.8541	0.0558
	STATE (2025) (Adduri et al., 2025)	0.003043	0.023399	0.5078	0.3911	0.3913	0.4340	0.1425	0.6795	0.0308
	scDFM (2026) (Yu et al., 2026)	0.002420	0.023003	0.6576	0.5295	0.7205	0.7341	0.5202	0.7697	0.0363
	Stack (2026) (Dong et al., 2026)	0.001777	0.019264	0.6927	0.5059	0.6749	0.6956	0.5142	0.7286	0.0608
	scCasFlow	**0.001416**	**0.014833**	**0.7643**	**0.6645**	**0.7692**	**0.7777**	**0.7248**	**0.8764**	**0.0846**

denote the ordered p-values. The corresponding BH-adjusted q-values were computed as

$$q_{(i)} = \min\left(1, \min_{j \geq i} \frac{m}{j} p_{(j)}\right). \tag{40}$$

Associations were considered significant when they satisfied both $q < 0.05$ and $|r_{k,t}| > 0.5$. The FDR threshold controls for multiple testing across all channel–TF associations, while the correlation threshold restricts the reported associations to those with $|r_{k,t}| > 0.5$.

The sign of $r_{k,t}$ reflects the orientation of the learned latent channel and is therefore not directly interpreted as TF activation or repression. We instead use $|r_{k,t}|$ to assess the strength of the association between a rewire channel and an inferred TF activity shift.

A.2.3 Time Scheduling Variant Details

scCasFlow uses two temporal variables: a rewire time $t_{\text{rewire}} \in [0, 1]$, which parameterizes progress in intermediate-response generation, and an expression time $t_{\text{express}} \in [0, 1]$, which parameterizes progress in transcriptomic latent generation. An inference schedule is defined as a monotone path:

$$\gamma : [0, 1] \to [0, 1]^2, \qquad \gamma(T) = \big(t_{\text{express}}(T), t_{\text{rewire}}(T)\big), \tag{41}$$

where $T \in [0, 1]$ denotes a shared integration variable, with $\gamma(0) = (0, 0)$ and $\gamma(1) = (1, 1)$. Different choices of γ encode different assumptions about the generation order.

Default Cascaded Path. The default scCasFlow inference scheme follows the ordered two-stage path

$$t_{\text{rewire}}(T) = \min(2T, 1), \qquad t_{\text{express}}(T) = \max(2T - 1, 0). \tag{42}$$

In the $(t_{\text{express}}, t_{\text{rewire}})$ plane, this corresponds to the rectilinear path

$$(0, 0) \to (0, 1) \to (1, 1),$$

under which intermediate-response generation is completed before expression generation begins. This path is equivalent to the two-stage inference procedure described in Algorithm 2.

Matched Smooth Schedules. To evaluate generation order under matched training and inference schedules, we use the following monotone family:

$$t_{\text{express}}(T) = T, \qquad t_{\text{rewire}}(T) = \frac{\alpha T}{1 + (\alpha - 1)T}, \qquad \alpha > 0. \tag{43}$$

This transformation continuously varies the relative progression of the two flows. When $\alpha > 1$, the intermediate-response coordinate advances faster than the expression coordinate, corresponding to an intermediate-leading schedule. When $\alpha = 1$, the two coordinates progress simultaneously along $t_{\text{rewire}} = t_{\text{express}} = T$. When $\alpha < 1$, the expression coordinate advances faster, corresponding to an expression-leading schedule.

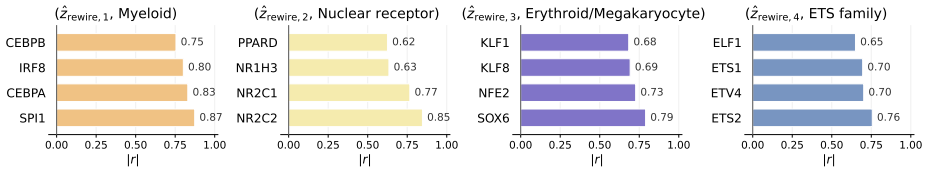

Figure S1: Top transcription factors (TFs) associated with selected rewire-phase latent channels. The bar plots show the absolute Pearson correlation ($|r|$) between predicted latent-channel activations ($\hat{z}_{\text{rewire},c}$) and TF activity shifts inferred from held-out perturbed expression across 31 unseen combinatorial perturbations. TF activity is estimated post hoc and is not used for intermediate-target construction, training, or inference. The displayed associations provide a complementary view of regulatory-associated structure in the learned intermediate representations rather than direct recovery of upstream regulatory states.

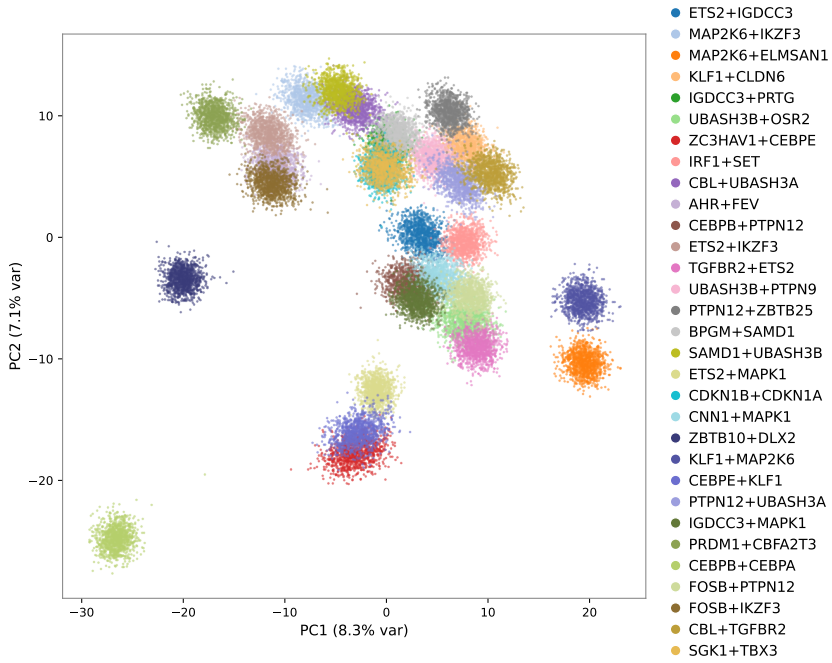

Figure S2: **PCA visualization of generated intermediate representations.** Representations are obtained after completing the Rewire-phase integration and projected onto the first two principal components.

B Additional Experimental Results

B.1 More Evaluation on the Norman holdout split

To provide a more granular understanding of model performance on the Norman holdout split, Table S2 presents a detailed evaluation separated into single- and double-perturbation settings. In the single-perturbation setting, scCasFlow achieves the best performance in Pearson Δ, DE Spearman LFC, and Overlap@50, and the second-best performance in DS, Pearson Δ_{20}, Pearson Δ_{50}, and DE Direction Match. In the double-perturbation setting, scCasFlow achieves the best performance across all reported metrics. It obtains a DS of 0.7643, Pearson Δ of 0.6645, and DE Direction Match of 0.8764, compared with the second-best values of 0.6927, 0.6082, and 0.8541, respectively. These results further support the ability of scCasFlow to generalize to held-out combinatorial perturbations in the Norman-Holdout setting.

Algorithm 1 Training scCasFlow

Input: Training control population $\mathcal{D}^{\mathrm{tr}}_{\mathrm{ctrl}}$; training perturbed populations $\{\mathcal{D}^{\mathrm{tr}}_{\mathbf{p}}\}_{\mathbf{p}\in\mathcal{P}_{\mathrm{tr}}}$; frozen encoder and tokenizer; $K = 128$, γ, ϵ, λ
Output: Vector-field parameters θ and decoder parameters ϕ

Compute $\boldsymbol{\mu}_{\mathrm{ctrl}}$ and fixed $\boldsymbol{\mu}_{\Delta}, \boldsymbol{\sigma}_{\Delta}$ from training cells and conditions only
for each training mini-batch **do**
  Initialize $\mathcal{L}_{\mathcal{B}} \leftarrow 0$ for mini-batch size B
  for mini-batch element $i = 1, \ldots, B$ **do**
    Sample $\mathbf{p} \in \mathcal{P}_{\mathrm{tr}}$ and $\mathbf{x}^{\mathrm{ctrl}} \sim \hat{q}^{\mathrm{tr}}_{\mathrm{ctrl}}$
    Construct $\mathbf{e}_{\mathbf{p}}$ with the condition encoder
    Draw a fresh K-cell cohort $\mathcal{S}^{K}_{\mathbf{p}} \subset \mathcal{D}^{\mathrm{tr}}_{\mathbf{p}}$ uniformly without replacement
    Recompute $\boldsymbol{\Delta}_{\mathbf{p},\mathcal{S}}$ and $\widetilde{\mathbf{r}}_{\mathbf{p},\mathcal{S}}$.
    Tokenize $\mathbf{x}^{\mathrm{ctrl}}$ to obtain $\mathbf{e}^{\mathrm{ctrl}}$
    Sample $\mathbf{z}_{\mathrm{rewire}}, \mathbf{z}_{\mathrm{express}} \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$ and $b \sim \mathrm{Bernoulli}(\gamma)$
    if $b = 1$ (intermediate mediator generation) **then**
      Sample $t_{\mathrm{rewire}} \sim \mathcal{U}(0, 1)$; set $t_{\mathrm{express}} = 0$ and $\mathbf{s} = \mathbf{z}_{\mathrm{express}}$
      Construct $\mathbf{h}$ from $\widetilde{\mathbf{r}}_{\mathbf{p},\mathcal{S}}$, predict $\widehat{\mathbf{v}}_{\mathrm{rewire}}$, and set $\ell_i = \ell_{\mathrm{rewire}}$
    else
      Independently sample $\mathbf{x}^{\mathrm{pert}} \sim \hat{q}^{\mathrm{tr}}_{\mathbf{p}}$ and tokenize it as $\mathbf{e}^{\mathrm{pert}}$
      Sample $t_{\mathrm{rewire}} \sim \mathcal{U}(1-\epsilon, 1)$ and $t_{\mathrm{express}} \sim \mathcal{U}(0, 1)$
      Construct $\mathbf{h}, \mathbf{s}$, predict $\widehat{\mathbf{v}}_{\mathrm{express}}$, and compute the explicit-Euler terminal estimate
      Set $\ell_i = \ell_{\mathrm{express}} + \lambda \ell_{\mathrm{recon}}$ as in Eq. 16
    end if
    Accumulate $\mathcal{L}_{\mathcal{B}} \leftarrow \mathcal{L}_{\mathcal{B}} + \ell_i / B$
  end for
  Update (θ, ϕ) using $\nabla_{\theta,\phi}\mathcal{L}_{\mathcal{B}}$
end for

Algorithm 2 Inference with scCasFlow

Input: Trained θ, ϕ; control profile $\mathbf{x}^{\mathrm{ctrl}}$; perturbation specification $\mathbf{p}$; $N_{\mathrm{rewire}} = N_{\mathrm{express}} = 20$. No perturbed profiles or cohort targets are inputs.
Output: Predicted expression $\widehat{\mathbf{x}}^{\mathrm{pert}}$

Encode $\mathbf{x}^{\mathrm{ctrl}}$ as $\mathbf{e}^{\mathrm{ctrl}}$ and construct the perturbation embedding $\mathbf{e}_{\mathbf{p}}$
Sample $\mathbf{h}, \mathbf{s} \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$
// Phase 1: Intermediate mediator generation
for $k = 0, \ldots, N_{\mathrm{rewire}} - 1$ **do**
  Set $t_{\mathrm{rewire}} = k/N_{\mathrm{rewire}}$, $t_{\mathrm{express}} = 0$
  $[\widehat{\mathbf{v}}_{\mathrm{rewire}}, -] \leftarrow \mathcal{V}_\theta(\mathbf{h}, \mathbf{s} \mid t_{\mathrm{rewire}}, t_{\mathrm{express}}, \mathbf{e}^{\mathrm{ctrl}}, \mathbf{e}_{\mathbf{p}})$
  Update $\mathbf{h} \leftarrow \mathbf{h} + \widehat{\mathbf{v}}_{\mathrm{rewire}}/N_{\mathrm{rewire}}$
end for
// Phase 2: Expression generation
for $k = 0, \ldots, N_{\mathrm{express}} - 1$ **do**
  Set $t_{\mathrm{rewire}} = 1$, $t_{\mathrm{express}} = k/N_{\mathrm{express}}$
  $[-, \widehat{\mathbf{v}}_{\mathrm{express}}] \leftarrow \mathcal{V}_\theta(\mathbf{h}, \mathbf{s} \mid t_{\mathrm{rewire}}, t_{\mathrm{express}}, \mathbf{e}^{\mathrm{ctrl}}, \mathbf{e}_{\mathbf{p}})$
  Update $\mathbf{s} \leftarrow \mathbf{s} + \widehat{\mathbf{v}}_{\mathrm{express}}/N_{\mathrm{express}}$
end for
return $\widehat{\mathbf{x}}^{\mathrm{pert}} \leftarrow \mathcal{D}_\phi(\mathbf{s})$

B.2 Additional TF-Association Analysis of Rewire-Phase Latent Channels

To further characterize the associations between rewire-phase latent representations ($\hat{\mathbf{z}}_{\mathrm{rewire}}$) and expression-derived transcription-factor (TF) activity, Figure S1 shows the TFs most strongly associated with four selected latent channels. TFs are ranked by the absolute Pearson correlation ($|r|$) between their inferred activity shifts and the corresponding channel activations across the 31 unseen combinatorial perturbations. As in the analysis reported in the main text, TF activities are inferred post hoc from held-out perturbed expression and are not used for intermediate-target construction, training, or inference.

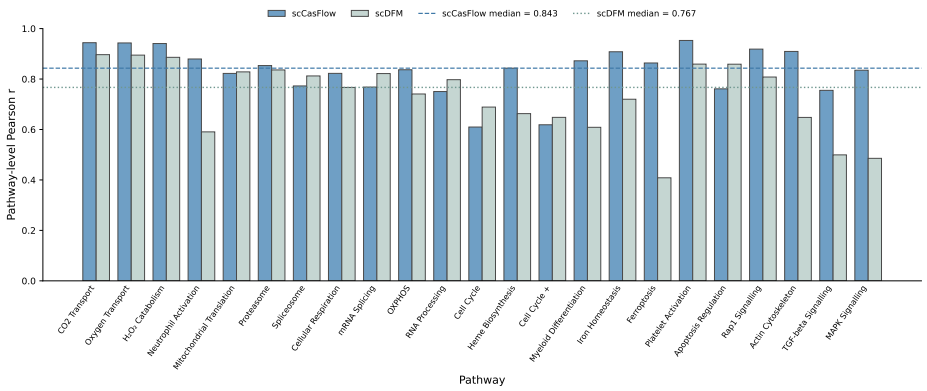

Figure S3: **Pathway-level response recovery across 23 predefined K562-relevant pathways.** Bars show the Pearson correlation between predicted and observed pathway-level responses across the held-out perturbations in the Norman-Additive split. Horizontal lines indicate the median correlations across pathways.

The selected channels exhibit different TF-association patterns. For $\hat{z}_{\mathrm{rewire},1}$, the top-associated TFs include SPI1, CEBPA, IRF8, and CEBPB, which are related to myeloid differentiation. For $\hat{z}_{\mathrm{rewire},2}$, the strongest associations include the nuclear receptors NR2C2, NR2C1, NR1H3, and PPARD. The top-associated TFs for $\hat{z}_{\mathrm{rewire},3}$ include SOX6, NFE2, KLF8, and KLF1, several of which are associated with erythroid or megakaryocytic programs. For $\hat{z}_{\mathrm{rewire},4}$, the top-associated TFs include ETS2, ETV4, ETS1, and ELF1, all belonging to the ETS family.

Taken together, these examples show that selected rewire-phase latent channels exhibit distinct association patterns with expression-derived TF activity scores, providing qualitative context for the systematic channel–TF associations reported in the main text. Because the TF activity scores are themselves inferred from transcriptomic profiles and no TF-based supervision enters the model, these associations support regulatory-associated structure in the intermediate representations rather than disentanglement or direct recovery of upstream regulatory programs.

B.3 Visualization of Intermediate Representations

To characterize the generated intermediate representations, we perform principal component analysis (PCA) on the representations obtained after completing the Rewire-phase integration. For each perturbation, we generate 1,024 stochastic samples by drawing independent initial noise vectors and integrating the Rewire flow while keeping the expression flow fixed ($t_{\mathrm{express}} = 0$). The resulting representations are projected onto the first two principal components. Figure S2 shows that samples associated with different perturbation conditions occupy distinct regions in the projected space. Because perturbation identity is explicitly provided as a conditioning input, this visualization is intended as a descriptive characterization of the intermediate representation space rather than evidence of regulatory disentanglement or direct recovery of upstream regulatory states.

B.4 Systematic Pathway-level Response Evaluation

Pathway-level response fidelity is evaluated on the Norman-Additive split using a panel of 23 K562-relevant pathways. Following the pathway-level analysis in the main text, we apply the same pathway-level aggregation to the 31 held-out perturbations and compute, for each pathway, the Pearson correlation between predicted and observed responses. We compare scCasFlow with scDFM under the same evaluation protocol.

As shown in Figure S3, scCasFlow achieves higher predicted–observed correlations than scDFM on 16 of the 23 pathways. The median Pearson correlation across the pathway panel is 0.843 for scCasFlow and 0.767 for scDFM, indicating that the predicted transcriptional responses preserve pathway-level patterns across a broader set of biological programs.

C ALGORITHM

The training and inference procedure for scCasFlow is detailed in Algorithm 1 and 2.

D LIMITATIONS AND FUTURE WORK

While scCasFlow improves predictive performance across the evaluated genetic and chemical perturbation benchmarks, several limitations remain. First, the intermediate target used by the current framework is a population-level response surrogate defined by perturbation-induced shifts in the embedding space of a frozen pretrained single-cell model, rather than a directly observed upstream regulatory state. Likewise, the TF associations reported in our analyses are derived post hoc from activity scores inferred from held-out transcriptomic profiles. The current results therefore do not establish that the rewire representation directly recovers causal regulatory mechanisms. Future work could incorporate matched multi-omic measurements, such as chromatin-accessibility or protein-abundance measurements, to construct or validate alternative intermediate representations and to examine whether richer molecular supervision improves perturbation-response prediction.

Our current evaluation is limited to two public single-cell perturbation benchmarks, Norman and ComboSciPlex, covering genetic and chemical perturbations. Although these benchmarks evaluate generalization to held-out genes and unseen perturbation combinations, broader validation across additional cell types, perturbation modalities, biological systems, and experimental platforms is needed to assess the robustness and generalizability of the framework. Future work could further extend *Rewire-then-Express* to temporal and multi-omic settings in which intermediate molecular responses can be measured more directly.

E BROADER IMPACT

Accurate prediction of cellular responses to perturbations can support hypothesis generation, experimental prioritization, and the study of cellular response programs. We present scCasFlow, a cascaded flow-matching framework that uses a biologically motivated ordered factorization, first predicting an intermediate response representation and then generating the transcriptomic response conditioned on it. Across the evaluated benchmarks, this formulation improves prediction under held-out perturbation settings involving unseen genes and perturbation combinations.

The computational efficiency of latent-space generation may make larger-scale *in silico* perturbation screening more tractable, potentially helping prioritize a smaller set of perturbations for experimental follow-up. The observed associations between rewire-phase latent channels and expression-derived TF activity scores may also provide auxiliary biological context for model predictions and motivate experimentally testable hypotheses. These model predictions and post hoc associations should not be interpreted as substitutes for experimental validation or as direct identification of causal regulatory mechanisms.

The framework may also be extended to multi-omic measurements, temporal perturbation data, or settings with explicit regulatory-network information. More broadly, our results motivate further investigation of ordered intermediate representations as inductive biases for generative models of cellular perturbation responses.