SICNN: Sparsity-induced Input Convex Neural Network for Optimal Transport

Peter Chen LC3826@COLUMBIA.EDU
Department of Mathematics, Columbia University
HKU Musketeers Foundation Institute of Data Science

Yue Xie YXIE21@HKU.HK
Qingpeng Zhang QPZHANG@HKU.HK
HKU Musketeers Foundation Institute of Data Science

Abstract

Optimal Transport (OT) theory seeks to find the map $T : X \to Y$ that transports the source measure X to the target measure Y with minimized cost $c(\mathbf{x}, T(\mathbf{x}))$ between $\mathbf{x}$ and its image $T(\mathbf{x})$. Building on the previous work of the Input Convex Neural Network (ICNN) OT solver [1, 13], and drawing inspiration from the concept of displacement-sparse maps [9], we introduce the sparsity penalty into the ICNN to promote sparsity in the displacement vectors $\Delta(\mathbf{x}) = T(\mathbf{x}) - \mathbf{x}$, making the resulting map with better interpretability. However, a side effect of increased sparsity is reduced feasibility, which means $T(X)$ may deviate more significantly from the actual target measure. In the low-dimensional setting, we propose a heuristic framework to balance the trade-off between the sparsity and feasibility of the map. This framework dynamically adjusts the sparsity-inducing intensity based on the evaluation of maps learned over different iterations. In the high-dimensional setting, we directly constrain the dimensionality of the displacement vectors, i.e., for $X \in \mathbb{R}^d$, $\forall \mathbf{x} \in X$, we enforce $\dim(\Delta(\mathbf{x})) \leq l$, where $l \ll d$. Among all maps that satisfy this constraint, we aim to find the most feasible map. We demonstrate that this formulation can be novelly solved using our heuristic adjustment framework without resorting to dimensionality reduction.

1. Introduction

Optimal transport (OT) aims to find a map T that efficiently transfers mass from one measure to another under the ground-truth cost. This technique has been applied in various machine-learning-based single-cell biology studies, as demonstrated in [6, 10, 11, 16, 22]. Compared to traditional methods of solving OT, such as Sinkhorn's algorithm [8], recent studies in single-cell biology have increasingly utilized Neural Optimal Transport (Neural OT) solvers. These solvers efficiently scale OT to large-scale and high-dimensional problems using continuous methods [12]. For instance, CellOT [6] addresses cell perturbation response by leveraging Neural OT, implemented via the Input Convex Neural Network (ICNN) framework [13].

While Neural OT solvers can produce feasible maps, the lack of sparsity in the displacement vector makes these maps less interpretable, i.e. it is challenging to understand the trend of how the source is transported to the target based on their feasible but complex results. Inspired by Cuturi

. This work was completed during Peter Chen's internship at HKU IDS and partially supported by the HKU IDS Research Grant.

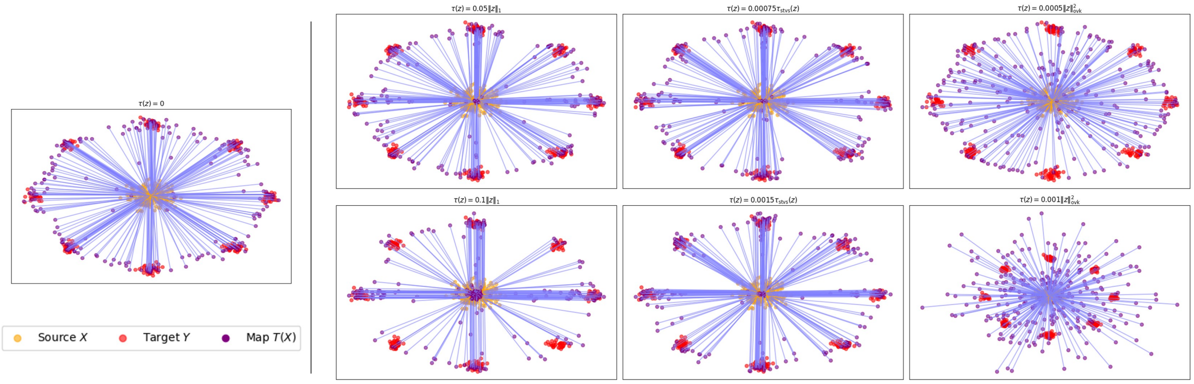

Figure 1: OT maps induced by various sparsity penalties $\tau(\cdot)$ at different intensity levels λ. The first column shows the map produced by ICNN OT without sparsity penalties. Subsequent columns compare maps with different sparsity penalties: the upper row illustrates the map trained with lower sparsity intensity, while the lower row shows the map trained with higher sparsity intensity. Details for these numerical experiments can found in Appendix G.

et al. [9]'s displacement-sparse map, which uses the cost function $c(\mathbf{x}, \mathbf{y}) := \frac{1}{2}\|\mathbf{x} - \mathbf{y}\|_2^2 + \lambda \cdot \tau$, where τ is a sparsity regulator and λ is a constant that controls the intensity of sparsity-inducing, we incorporate these sparsity penalties into the minimax formulation of the Kantorovich dual problem proposed by Makkuva et al. [13]. This minimax formulation re-parameterized Kantorovich dual potentials f and g into the convex function space, which can be then learned via ICNN training (we denote this method as ICNN OT [13] in the later discussion).

However, inducing sparsity in the OT map comes with a trade-off: as the sparsity level increases, the accuracy of the map's feasibility decreases, which means that the source distribution cannot be effectively transported to the target distribution. This effect is evident in Figure 1, where at higher intensity levels, more points mapped by T are clustered around the center of the source distribution. To tackle this issue, we create a dynamic framework to adjust the sparsity intensity, offering two models to solve OT tasks in both low-dimensional and high-dimensional spaces.

Discussion of more related works is in Appendix A. Our contributions is summarized below:

- We embed the sparsity penalty functions into the ICNN OT framework. For low-dimensional spaces, we apply three sparsity penalties (ℓ_1-norm, STVS, and OVK) summarized in [9]. For high-dimensional spaces, we introduce a novel smoothed ℓ_0-norm as penalty.
- For low-dimensional tasks, we introduce a novel framework within the ICNN training process that heuristically adjusts the sparsity intensity λ, allowing users to balance the trade-off between the feasibility and sparsity of the resulting map.
- For high-dimensional tasks, we directly constrain the dimensionality of the displacement vectors, aiming to find the most feasible map among those that satisfy these constraints. We find that this formulation could be solved via our novel ICNN framework. We also propose another theoretical "ground-truth" solution alongside the solution generated by our new ICNN framework. We have included the entire implementation in this anonymous repository.

2. Background in Optimal Transport

Given two distributions P and Q in $\mathbb{R}^d$ with a quadratic transportation cost, the Monge problem seeks to find the transport map $T : \mathbb{R}^d \to \mathbb{R}^d$ that minimizes the transportation cost:

$$T^* = \operatorname*{arg\,min}_{T:T_\# P=Q} \frac{1}{2}\mathbb{E}_{X\sim P}\|T(X) - X\|^2. \tag{1}$$

Here, T^* is the optimal transport map among all maps T that push forward P to Q. However, solving (1) is challenging because the set of feasible maps T may not be convex or might not exist at all. To address this, Kantorovich relaxed the problem by allowing mass splitting, replacing the direct map T with a coupling π between P and Q:

$$W_2^2(P,Q) = \inf_{\pi\in\Pi(P,Q)} \frac{1}{2}\mathbb{E}_{(X,Y)\sim\pi}\|X - Y\|^2, \tag{2}$$

where $\Pi(P,Q)$ is the set of all couplings (or transport plans) between P and Q. This relaxation makes the problem convex and solvable via linear programming, with the optimal value being the squared 2-Wasserstein distance. To recover the map T from a coupling Π, one can use Kantorovich's dual formulation[19, Theorem 1.3]:

$$W_2^2(P,Q) = \sup_{(f,g)\in\Phi_c} \left(\mathbb{E}_P[f(X)] + \mathbb{E}_Q[g(Y)]\right), \tag{3}$$

where $\Phi_c = \left\{(f,g) \in L^1(P) \times L^1(Q) : f(x) + g(y) \le c(x,y)\right\}$ is the constraint space for the dual potentials f and g, with $L^1(P)$ denoting the set of integrable functions with respect to P, defined as $\{f : f$ is measurable and $\int |f|\, dP < \infty\}$, and $c(x,y)$ represents the transportation cost. Given the quadratic cost $c(x,y) = \frac{1}{2}\|x-y\|^2$ and under the saturated condition $f(x) + g(y) = \frac{1}{2}\|x-y\|^2$, map T can be recovered as [5]:

$$T(x) = x - \nabla f(x) \tag{4}$$

Dual formulation (3) can be further re-parameterized into a minimax formulation [13] (details are provided in the Appendix B):

$$W_2^2(P,Q) = \sup_{\substack{f\in\texttt{CVX}(P)\\ f^*\in L^1(Q)}} \inf_{g\in\texttt{CVX}(Q)} \mathcal{V}_{P,Q}(f,g) + C_{P,Q}, \tag{5}$$

where $\mathcal{V}_{P,Q}(f,g) = -\mathbb{E}_P[f(X)] - \mathbb{E}_Q\left[\langle Y, \nabla g(Y)\rangle - f(\nabla g(Y))\right]$, $C_{P,Q} = \frac{1}{2}\mathbb{E}[\|X\|^2 + \|Y\|^2]$, $f^*(y) = \sup_{x\in\mathbb{R}^n}(\langle x,y\rangle - f(x))$ denotes the convex conjugate of f, and $\texttt{CVX}(P)$ denotes the set of all convex functions in $L^1(P)$. The new minimax formulation parameterized f and g into the convex space under an separate alternative optimization scheme, which can be learned via ICNN. Note that if optimal solution (f_0, g_0) exists, then optimal transport map T can be recovered via $\nabla g_0(\cdot)$[13, Theorem 3.3]. Technical details of this implementation can be referred at Appendix C.

3. Inducing Sparsity towards Displacement Vector

Our goal is to bring sparsity to the displacement vector $\Delta(\mathbf{x}) = T(\mathbf{x}) - \mathbf{x}$ from the map T learned via ICNN, making the result with better interpretability. We introduce the sparsity penalty to the minimax formulation (5), which makes the squared 2-Wasserstein biased:

$$\widetilde{W_2^2}(P,Q) = C_{P,Q} + \sup_{\substack{f\in\mathtt{CVX}(P)\\ f^*\in L^1(Q)}} \inf_{g\in\mathtt{CVX}(Q)} \mathcal{V}_{P,Q}(f,g) + \lambda \int_{\mathbb{R}^d} \tau(\nabla g(y) - y)\, dQ, \tag{6}$$

where $\nabla g(y) - y$ represents displacement vector derived from (4), $\tau(\cdot)$ is the sparsity penalty for displacement vectors, and λ is a parameter that controls the intensity of sparsity-inducing.

We first introduce three theorems to show the theoretical feasibility of formulation (6) before delving into other details.

Theorem 1 (Convergence of the Wasserstein Distance) *Let J_0 denote the Wasserstein distance as defined in equation (5), and let J_λ denote the biased Wasserstein distance as defined in equation (6), with λ representing the sparsity-inducing intensity. Then, it holds that:*

$$\lim_{\lambda\to 0} J_\lambda = J_0.$$

Theorem 2 (Convergence of Dual Potentials for Convex τ) *Let (g_0^*, f_0^*) be the optimal solution of the optimization problem defined in equation (5), and let $(g_\lambda^*, f_\lambda^*)$ be the optimal solution to equation (6) for a given $\lambda \geq 0$ and a convex regulator τ. Then, as $\lambda \to 0$, it holds that:*

$$(g_\lambda^*, f_\lambda^*) \to (g_0^*, f_0^*).$$

Note that $(g_\lambda^*, f_\lambda^*)$ may not converge to a unique (g_0^*, f_0^*) unless the objective function is strongly concave-strongly convex. $(g_\lambda^*, f_\lambda^*)$ would converge to a set of saddle points (g_0^*, f_0^*) if objective function is only concave-convex.

Theorem 3 (Monotonicity of Regularization for Convex τ) *Let $(g_{\lambda_1}^*, f_{\lambda_1}^*)$ and $(g_{\lambda_2}^*, f_{\lambda_2}^*)$ be the optimal solutions to the optimization problem defined in equation (6) for $\lambda = \lambda_1$ and $\lambda = \lambda_2$, respectively, where $0 \leq \lambda_1 < \lambda_2$, and let τ be a convex regulator. Then, the following inequality holds:*

$$\int_{\mathbb{R}^d} \tau(\nabla g_{\lambda_1}^*(y) - y)\, dQ \geq \int_{\mathbb{R}^d} \tau(\nabla g_{\lambda_2}^*(y) - y)\, dQ.$$

Theorem 1 establishes that the formulation remains unbiased in the limit as $\lambda \to 0$ for any of the four regulators we used. Theorem 2 demonstrates the convergence of the dual potentials specifically for convex regulators. Theorem 3 shows the monotonicity of the map's sparsity level as λ increases. However, we empirically observe that our non-convex regulators also exhibit this property, as discussed in a later section. Proofs for these theorems are provided in the appendix D and E.

3.1. Sparsity-inducing Penalties τ

We introduce four functions for the sparsity-inducing penalty τ that promote sparsity in the displacement vectors: the ℓ_1-norm, the Vanishing Shrinkage STVS (τ_{stvs}), the k-overlap norm ($\|\cdot\|_{\text{ovk}}$), and the smoothed ℓ_0-norm (designed for high dimensional cases). For a detailed introduction to these penalties, please refer to Appendix F. Examples of ℓ_1, τ_{stvs}, and $\|\cdot\|_{\text{ovk}}$ used as penalty functions are shown in Figure 1. Among these three penalties, we find that $\|\cdot\|_{\text{ovk}}$ does not yield the expected sparsity-inducing results due to its inability to converge (referred to Appendix G), likely because of its quadratic nature.

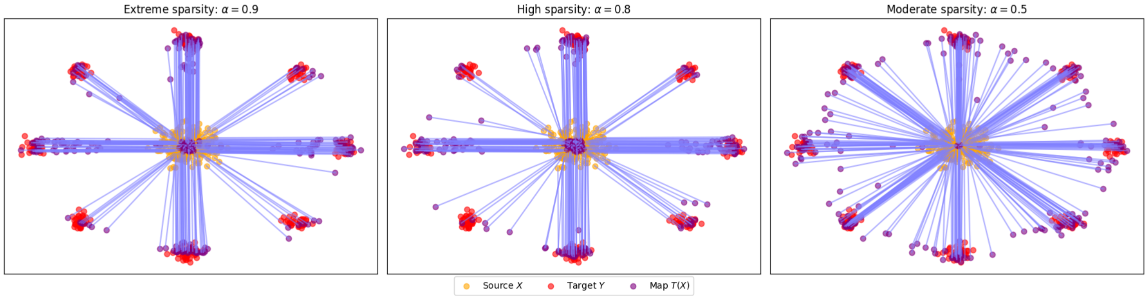

Figure 2: OT maps learned via ICNN with respect different α values in Eval from (7). τ_{stvs} is chosen as penalty function; From right to left, sparsity-inducing intensity increases with higher α. Example simulation can be found in Figure 7.

3.2. Low Dimensional Space: Dynamic Adjustment to Sparsity-inducing intensity λ

In Section 3.1, we discuss using a constant λ to learn a sparse map via ICNN. However, a constant intensity presents certain challenges: if the intensity is too low, the resulting map will not be sufficiently sparse; conversely, if the intensity is too high, the resulting map will be less accurate in feasibility (i.e. $T(X)$ deviated more from the actual Y).

In this section, we propose a new framework that dynamically adjusts λ to optimize a goal function that balances sparsity and feasibility during the ICNN training process. For the map T_n learned at the n-th iteration of ICNN training, we define its sparsity and feasibility level through two metrics, Spa (Sparsity) and Err (Error), where

$$\text{Spa} = \sum_{x \in X} \tau(T_n(x) - x), \quad \text{Err} = D(T_n(X), Y).$$

We use the corresponding sparsity metric, τ, during training, while measuring the error through the divergence D between the mapped distribution and the actual target distribution. To capture this geometric divergence, we approximate the source and target distributions, X and Y, as finite sets of sampled points, and use the Wasserstein distance to quantify the divergence between them. However, due to the computational inefficiency of the Wasserstein distance, we accelerate the training process by approximating it with the sliced Wasserstein distance [4]. It is important to normalize both metrics to the same numerical scale. Ultimately, we construct the evaluation function as:

$$\text{Eval} = \alpha \cdot \text{Spa} + (1 - \alpha) \cdot \text{Err}, \quad \alpha \in [0, 1], \tag{7}$$

where α is a pre-determined parameter. A higher value of α leads to a higher sparsity-inducing intensity. The objective is to find the λ that minimizes Eval. Since λ is not explicitly related to Eval, we use the heuristic search by embedding a simulated annealing framework into ICNN. We heuristically adjust λ after every specific number of iterations (to give the ICNN sufficient time to smoothly adjust the map with the updated λ). Examples are illustrated in Figure 2, and specific details of this implementation are provided in Appendix H.

3.3. High Dimensional Space: Direct Displacement Vector Dimensionality Constraint

In high-dimensional settings, instead of using a sparsity metric, we impose a direct constraint on the dimensionality of the displacement vectors. Given $P, Q \in \mathbb{R}^d$, our goal is to find an optimal

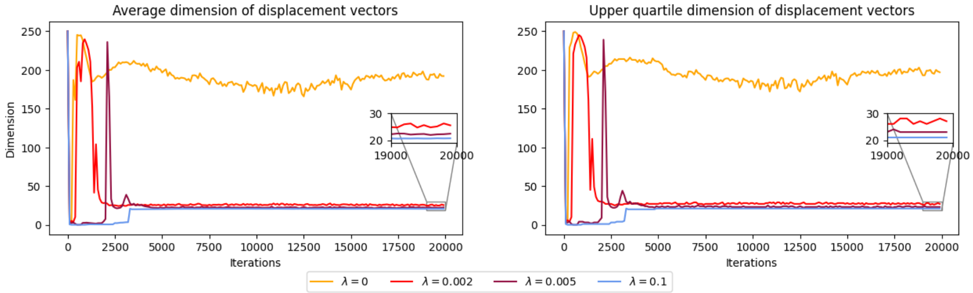

Figure 3: In $\mathbb{R}^d$, where $d = 250$, we simulated sc-RNA perturbations under a drug simulation scenario. In this setting, 20 genes are directly affected with significant changes, while other genes are perturbed within the noise level. An OT map is learned between the cell in original group and treated genes with a constant λ. Due to precision limitations of the ICNN, we count any entry of $\Delta(\mathrm{x})$ that larger than 1 as an extra dimension. The average and upper quartile of $\dim(\Delta(\mathrm{x}))$ are presented above. Note that ICNN can occasionally generate displacement vectors with dimensions significantly higher than the rest. To mitigate this, we recommend applying a quantile-based or average dimension constraint rather than imposing a strict upper limit on all vectors. A strict constraint could lead to undesirable outcomes, where most vectors have dimensions much lower than expected.

transport map T such that all displacement vectors $\Delta(\mathbf{x}) = T(\mathbf{x}) - \mathbf{x}$ have a dimensionality less than l, where $l \ll d$. Among all maps satisfying this constraint, we seek the one that minimizes the divergence between $T(X)$ and Y. This problem can be formulated as follows:

$$\begin{aligned} \min_{T} \quad & \frac{1}{2}\mathbb{E}_{X\sim P} \left\|T(X) - X\right\|^2 + \beta D(T(X), Y) \\ \text{s.t.} \quad & \dim(T(x_i) - x_i) \leq l, \quad l \ll d, \quad \forall x_i \in X. \end{aligned} \tag{8}$$

By introducing a divergence regularization term $D(T(X), Y)$ with weighting parameter β, we relax the requirement that the source measure P must be exactly transported to the target measure Q (i.e., $T_{\#}P = Q$). This formulation is indeed challenging to solve with traditional methods due to the direct constraint on the dimensionality of the vectors. However, our new heuristic adjustment framework based on ICNN can efficiently address this problem. We also provide a theoretical method to solve (8), though it incurs a high computational cost, as discussed in Appendix I.

We now introduce our ICNN-based solution. In contrast to lower-dimensional cases, we can directly apply the smoothed ℓ_0-norm to control the dimensionality of the displacement vectors. Although the smoothed ℓ_0-norm is not a convex penalty, we empirically observe that increasing λ enhances sparsity, as shown in Figure 3. Therefore, our strategy to adjust λ becomes straightforward: we initially increase λ until the dimensionality requirement is met; then, we decrease λ to minimize $D(T(X), Y)$ while maintaining the dimensionality constraint. For further details on the algorithm, please refer to Appendix J.

3.4. Experiment Setting

Numerical experiments analysis for the toy data in Figures 1 and 2 can be found in Appendix G. For the higher-dimensional experiments, we construct synthesized single-cell RNA (sc-RNA) perturbation data by simulating changes in gene expression under drug stimulation. OT is used to match the cells between the original group and treated group via their gene expressions. In our synthesized data, we introduce a significant change in a small subset of genes truly affected by the drug while adding noise to other genes. Our objective is to learn an OT map that accurately captures the transport among the most significantly affected genes.

Compared to methods such as using PCA or sliced-Wasserstein method [7, 15] to reduce the problem to a lower dimension and then run the OT solver, where the learned map may lack interpretability in the original space, our method maintains the interpretability of the map in the original high-dimensional space. Details of experiment and hyper-parameter setup can be found in Appendix K and Figure 3.

Due to the page limit, we put the conclusion part to Appendix L.

References

[1] Brandon Amos, Lei Xu, and J. Zico Kolter. Input convex neural networks. In *Proceedings of the 34th International Conference on Machine Learning*, volume 70 of *Proceedings of Machine Learning Research*, pages 146–155. PMLR, 2017.

[2] Andreas Argyriou, Rina Foygel, and Nathan Srebro. Sparse prediction with the k-support norm. In F. Pereira, C.J. Burges, L. Bottou, and K.Q. Weinberger, editors, *Advances in Neural Information Processing Systems*, volume 25. Curran Associates, Inc., 2012. URL https://proceedings.neurips.cc/paper_files/paper/2012/file/99bcfcd754a98ce89cb86f73acc04645-Paper.pdf.

[3] Mathieu Blondel, Vivien Seguy, and Antoine Rolet. Smooth and sparse optimal transport. In *Proceedings of the 21st International Conference on Artificial Intelligence and Statistics (AISTATS)*, volume 84 of *Proceedings of Machine Learning Research*, pages 880–889. PMLR, 2018. URL http://proceedings.mlr.press/v84/blondel18a/blondel18a.pdf.

[4] Nicolas Bonneel, Julien Rabin, Gabriel Peyré, and Hanspeter Pfister. Sliced and radon wasserstein barycenters of measures. *Journal of Mathematical Imaging and Vision*, 51(1):22–45, 2015.

[5] Yann Brenier. Polar factorization and monotone rearrangement of vector-valued functions. *Communications on Pure and Applied Mathematics*, 44(4):375–417, 1991.

[6] Cédric Bunne, Sebastian G. Stark, Gregor Gut, et al. Learning single-cell perturbation responses using neural optimal transport. *Nature Methods*, 20:1759–1768, 2023. doi: 10.1038/s41592-023-01969-x.

[7] Antoine Collas, Titouan Vayer, Rémi Flamary, and Arnaud Breloy. Entropic wasserstein component analysis, 2023. URL https://arxiv.org/abs/2303.05119.

[8] Marco Cuturi. Sinkhorn distances: Lightspeed computation of optimal transport. In C.J. Burges, L. Bottou, M. Welling, Z. Ghahramani, and K.Q. Weinberger, editors, *Advances in Neural Information Processing Systems*, volume 26. Curran Associates, Inc., 2013. URL https://proceedings.neurips.cc/paper_files/paper/2013/file/af21d0c97db2e27e13572cbf59eb343d-Paper.pdf.

[9] Marco Cuturi, Michal Klein, and Pierre Ablin. Monge, bregman and occam: Interpretable optimal transport in high-dimensions with feature-sparse maps. In *Proceedings of the 40th International Conference on Machine Learning*, volume 202 of *Proceedings of Machine Learning Research*, pages 6671–6682. PMLR, 2023.

[10] Pinar Demetci, Ryan Santorella, Bjorn Sandstede, William S. Noble, and Ritambhara Singh. Scot: single-cell multi-omics alignment with optimal transport. *Journal of Computational Biology*, 29:3–18, 2022. doi: 10.1089/cmb.2021.0295.

[11] Geert-Jan Huizing, Gabriel Peyré, and Laura Cantini. Optimal transport improves cell–cell similarity inference in single-cell omics data. *Bioinformatics*, 38(8):2169–2177, 2022. doi: 10.1093/bioinformatics/btac084. URL `https://doi.org/10.1093/bioinformatics/btac084`.

[12] Alexander Korotin, Lingxiao Li, Aude Genevay, Justin Solomon, Alexander Filippov, and Evgeny Burnaev. Do neural optimal transport solvers work? a continuous wasserstein-2 benchmark. In *Advances in Neural Information Processing Systems*, NIPS '21. Curran Associates Inc., 2021.

[13] Ashok Makkuva, Amirhossein Taghvaei, Sewoong Oh, and Jason Lee. Optimal transport mapping via input convex neural networks. In *Proceedings of the 37th International Conference on Machine Learning*, volume 119 of *Proceedings of Machine Learning Research*, pages 6672–6681. PMLR, 2020.

[14] G. Hosein Mohimani, Massoud Babaie-Zadeh, and Christian Jutten. Fast sparse representation based on smoothed ℓ_0 norm. In Mike E. Davies, Christopher J. James, Samer A. Abdallah, and Mark D. Plumbley, editors, *Independent Component Analysis and Signal Separation*, pages 389–396, Berlin, Heidelberg, 2007. Springer Berlin Heidelberg.

[15] Julien Rabin, Gabriel Peyré, Julie Delon, and Marc Bernot. Wasserstein barycenter and its application to texture mixing. In Alfred M. Bruckstein, Bart M. ter Haar Romeny, Alexander M. Bronstein, and Michael M. Bronstein, editors, *Scale Space and Variational Methods in Computer Vision*, pages 435–446, Berlin, Heidelberg, 2012. Springer Berlin Heidelberg. ISBN 978-3-642-24785-9.

[16] G. Schiebinger, J. Shu, M. Tabaka, B. Cleary, V. Subramanian, A. Solomon, J. Gould, S. Liu, S. Lin, P. Berube, L. Lee, J. Chen, J. Brumbaugh, P. Rigollet, K. Hochedlinger, R. Jaenisch, A. Regev, and E. S. Lander. Optimal-transport analysis of single-cell gene expression identifies developmental trajectories in reprogramming. *Cell*, 176(4):928–943.e22, 2019. doi: 10.1016/j.cell.2019.01.006.

[17] Amandine Schreck, Gersende Fort, Sylvain Le Corff, and Eric Moulines. A shrinkage-thresholding metropolis adjusted langevin algorithm for bayesian variable selection. *IEEE Journal of Selected Topics in Signal Processing*, 10(2):366–375, 2015.

[18] Mathieu Blondel Tianlin Liu, Joan Puigcerver. Sparsity-constrained optimal transport. In *Proceedings of the Eleventh International Conference on Learning Representations (ICLR)*, 2023. URL `https://openreview.net/forum?id=yHY9NbQJ5BP`.

[19] Cédric Villani. *Topics in Optimal Transportation*, volume 58. American Mathematical Society, 2003.

[20] John von Neumann. Zur theorie der gesellschaftsspiele. *Mathematische Annalen*, 100:295–320, 1928. doi: 10.1007/BF01448847.

[21] Yuanhao Wang and Jian Li. Improved algorithms for convex-concave minimax optimization. In H. Larochelle, M. Ranzato, R. Hadsell, M.F. Balcan, and H. Lin, editors, *Advances in Neural Information Processing Systems*, volume 33, pages 4800–4810. Curran Associates, Inc., 2020. URL https://proceedings.neurips.cc/paper_files/paper/2020/file/331316d4efb44682092a006307b9ae3a-Paper.pdf.

[22] S. Zhang, A. Afanassiev, L. Greenstreet, T. Matsumoto, and G. Schiebinger. Optimal transport analysis reveals trajectories in steady-state systems. *PLoS Computational Biology*, 17(7): e1009466, 2021. doi: 10.1371/journal.pcbi.1009466.

Appendix A. Related Works

While many recent works have related sparsity to OT, the notion of sparsity they address typically pertains to the complexity of the transportation plan matrix or the dual solutions. The idea of considering the sparsity of the displacement vector, however, is a novel concept introduced by Cuturi et al. [9] in 2023. This foundational work inspires us to develop neural OT solvers that not only exhibit these traits but also leverage them to address more complex displacement-related sparsity problems. Below, we introduce several related works focusing on other aspects of sparsity in OT.

Sparsity-Constrained Optimal Transport. Liu et al. [18] proposed a regularization method to control the column cardinality of the transportation plan matrix, achieving strong dual results. This regularization leads to greater sparsity and reduced complexity in the transportation plan. However, converting the transportation plan to the map typically depends on the characteristics of the rows, which is not the focus of this study. Moreover, when converting the plan to displacement vectors using methods such as Barycenters, the constrained cardinality of entries in the plan does not necessarily translate to the dimensionality of the displacement vector.

Smooth and Sparse Optimal Transport. Blondel et al. [3] propose a regularization method of using the squared 2-norm to handle the dense and complex transportation plans produced by the entropic regularization solution via Sinkhorn's algorithm.

Appendix B. Details for Minimax Formulation (5)

The constraint in equation (3),

$$f(x) + g(y) \leq \frac{1}{2}\|x - y\|_2^2,$$

can be reparameterized as:

$$\left[\frac{1}{2}\|x\|_2^2 - f(x)\right] + \left[\frac{1}{2}\|y\|_2^2 - g(y)\right] \geq \langle x, y \rangle.$$

By reparameterizing f and g using $\frac{1}{2}\|x\|_2^2 - f(x)$ and $\frac{1}{2}\|y\|_2^2 - g(y)$, respectively, and substituting these into the dual formulation (3), we obtain:

$$W_2^2(P, Q) = C_{P,Q} - \inf_{(f,g)\in\widetilde{\Phi}_c} \left\{\mathbb{E}_P[f(X)] + \mathbb{E}_Q[g(Y)]\right\},$$

where

$$\widetilde{\Phi}_c \triangleq \{(f, g) : f(x) + g(y) \geq \langle x, y \rangle\},$$

and $C_{P,Q} = \frac{1}{2}\mathbb{E}[\|X\|_2^2 + \|Y\|_2^2]$ is a constant. Makkuva et al. [13, Theorem 3.3] have shown that this reparameterized formulation can be further transformed into the minimax formulation (5) and solved via ICNN.

Appendix C. Background in Input Convex Neural Network (ICNN)

Amos et al. [1] introduced Input Convex Neural Networks (ICNNs) to learn convex functions by finding a set of parameters, $\{W_i^{(y)}, W_i^{(z)}\}$, during each iteration to optimize a specific objective function.

Given an initial input y, the neural network learns the convex function z_n through the following model:

$$z_{i+1} = g_i\left(W_i^{(z)} z_i + W_i^{(y)} y + b_i\right), \quad f(y;\theta) = z_n,$$

where z_i denotes the layer activations, $\theta = \left\{W_{0:k-1}^{(y)}, W_{1:k-1}^{(z)}, b_{0:k-1}\right\}$ represents the parameters (with $W_{0:k-1}^{(y)}$ and $W_{1:k-1}^{(z)}$ as weight matrices and $b_{0:k-1}$ as the biased term), and g_i is non-linear activation functions. In each iteration, a small batch is selected from both the source and target, and the ICNN learns θ_f and θ_g within its $\texttt{ICNN}(\mathbb{R}^d)$ space to optimize the empirical counterpart of equation (5).

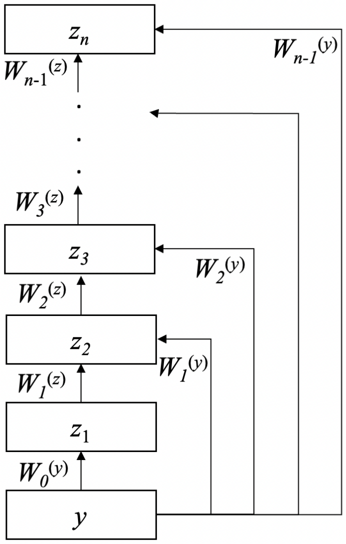

Figure 4: Input convex neural network (ICNN) architecture.

To maintain the convexity of the network, the parameters θ must be non-negative. Makkuva et al. demonstrates the feasibility of adding bounds to θ during the training process [13, Remark 3.4] ; This implementation could make that θ_f and θ_g can be learned from a compact function space. Further details of the algorithm can be found in [13, Algorithm 1].

Appendix D. Proof of Theorem 1 and 2.

To rigorously prove Theorem 1, we require following supporting theorem and lemmas.

Lemma 4 *Let $\tau(\cdot)$ be one of the four sparsity penalties introduced in Section 3.1. Consider λ, the sparsity-inducing intensity, and $\nabla g(y) - y$, the displacement vector, as given in equation (6). Then, it holds that:*

$$\lim_{\lambda \to 0} \lambda \int_{\mathbb{R}^d} \tau(\nabla g(y) - y)\, dQ = 0.$$

Sketch Proof of Lemma 4: To prove this lemma, we must show that $\tau(\cdot)$ is bounded. We assume that the source and target measures are separated by a finite distance, i.e., $\nabla g(y) - y$ is bounded. Indeed, each of the four sparsity penalties is bounded as follows:

- The smoothed ℓ_0-norm is bounded by the dimensionality of the source and target measures.
- The ℓ_1-norm and the $\|\cdot\|_{\text{ovk}}$ are bounded by the length of displacement vectors.
- The sparsity penalty τ_{stvs} is bounded by functions such as $\operatorname{asinh}(\cdot)$ and $e^{-(\cdot)}$.

Given these bounds, we can conclude that Lemma 4 holds for all four sparsity penalties used in our analysis.

Theorem 5 (von Neumann's Minimax Theorem [20]) *Let $X \subset \mathbb{R}^n$ and $Y \subset \mathbb{R}^m$ be two compact and convex sets. Suppose $f : X \times Y \to \mathbb{R}$ is a continuous function that is concave in x for each fixed y and convex in y for each fixed x (concave-convex). Then, the following equality holds:*

$$\max_{x\in X} \min_{y\in Y} f(x,y) = \min_{y\in Y} \max_{x\in X} f(x,y).$$

Moreover, its optimal solution (x^*, y^*) satisfies the following saddle point inequality:

$$f(x,y^*) \le f(x^*,y^*) \le f(x^*,y)$$

In the context of ICNN, we can relax the requirement of having two compact sets to a single compact space, $\texttt{ICNN}(\mathbb{R}^d)$.

Lemma 6 *For the expression defined in equation (5), given $f \in \texttt{CVX}(P)$ and $g \in \texttt{CVX}(Q)$, the function*

$$\mathcal{V}_{P,Q}(f,g) = -\mathbb{E}_P[f(X)] - \mathbb{E}_Q\left[\langle Y, \nabla g(Y)\rangle - f(\nabla g(Y))\right]$$

is a concave-convex function.

Proof of Lemma 6. To prove that $\mathcal{V}_{P,Q}(f,g)$ is a concave-convex function, we first show that, for a fixed g, $\mathcal{V}_{P,Q}(f,g)$ is concave in f.

Consider any two functions $f_1, f_2 \in \texttt{CVX}(P)$ and any $\alpha \in [0,1]$. We aim to show that:

$$\mathcal{V}_{P,Q}(\alpha f_1 + (1-\alpha) f_2, g) \ge \alpha \mathcal{V}_{P,Q}(f_1, g) + (1-\alpha)\mathcal{V}_{P,Q}(f_2, g). \tag{9}$$

To demonstrate this, note that by linearity:

$$\begin{aligned}
\mathcal{V}_{P,Q}(\alpha f_1 + (1-\alpha) f_2, g) &= -\mathbb{E}_P[(\alpha f_1 + (1-\alpha) f_2)(X)] \\
&\quad - \mathbb{E}_Q\left[\langle Y, \nabla g(Y)\rangle - (\alpha f_1 + (1-\alpha) f_2)(\nabla g(Y))\right] \\
&= -\alpha \mathbb{E}_P[f_1(X)] - (1-\alpha)\mathbb{E}_P[f_2(X)] \\
&\quad + \alpha \mathbb{E}_Q\left[f_1(\nabla g(Y))\right] + (1-\alpha)\mathbb{E}_Q\left[f_2(\nabla g(Y))\right] \\
&\quad - \mathbb{E}_Q\left[\langle Y, \nabla g(Y)\rangle\right] \\
&= \alpha \mathcal{V}_{P,Q}(f_1, g) + (1-\alpha)\mathcal{V}_{P,Q}(f_2, g).
\end{aligned}$$

This establishes that $\mathcal{V}_{P,Q}(f,g)$ is concave in f for a fixed g.

Next, we show that for a fixed f, $\mathcal{V}_{P,Q}(f,g)$ is convex in g. Consider any $g_1, g_2 \in \texttt{CVX}(Q)$ and any $\alpha \in [0,1]$. We need to prove that:

$$\mathcal{V}_{P,Q}(f, \alpha g_1 + (1-\alpha) g_2) \le \alpha \mathcal{V}_{P,Q}(f, g_1) + (1-\alpha)\mathcal{V}_{P,Q}(f, g_2). \tag{10}$$

By definition, we have:

$$\begin{aligned}\mathcal{V}_{P,Q}(f, \alpha g_1 + (1-\alpha) g_2) &= -\mathbb{E}_P[f(X)] \\ &\quad - \mathbb{E}_Q\left[\langle Y, \alpha \nabla g_1(Y) + (1-\alpha)\nabla g_2(Y)\rangle - f\big(\alpha \nabla g_1(Y) + (1-\alpha)\nabla g_2(Y)\big)\right] \\ &= -\mathbb{E}_P[f(X)] - \mathbb{E}_Q\left[\langle Y, \alpha \nabla g_1(Y)\rangle + \langle Y, (1-\alpha)\nabla g_2(Y)\rangle\right] \\ &\quad + \mathbb{E}_Q\left[f\big(\alpha \nabla g_1(Y) + (1-\alpha)\nabla g_2(Y)\big)\right] \\ &\leq -\mathbb{E}_P[f(X)] - \mathbb{E}_Q\left[\langle Y, \alpha \nabla g_1(Y)\rangle + \langle Y, (1-\alpha)\nabla g_2(Y)\rangle\right] \\ &\quad + \mathbb{E}_Q\left[\alpha f(\nabla g_1(Y)) + (1-\alpha) f(\nabla g_2(Y))\right] \quad (\because f \text{ is convex}).\end{aligned}$$

□

Combining the terms, we obtain the desired inequality (10). Therefore, by combining inequalities (9) and (10), we conclude that $\mathcal{V}_{P,Q}(f, g)$ is indeed a concave-convex function. With Lemma 4, Theorem 5, and Lemma 6, we can then prove Theorem 1.

Proof of Theorem 1. For brevity, let $S(g) = \int_{\mathbb{R}^d} \tau(\nabla g(y) - y)\, dQ$. We first aim to prove that $J_\lambda \geq J_0$. This follows from the non-negativity of $\tau(\cdot)$ and λ, which implies:

$$\mathcal{V}_{P,Q}(f, g) + \lambda S(g) \geq \mathcal{V}_{P,Q}(f, g).$$

Taking the infimum over g on both sides and then the supremum over f, we obtain:

$$\sup_{\substack{f \in \mathtt{CVX}(P) \\ f^* \in L^1(Q)}} \inf_{g \in \mathtt{CVX}(Q)} \left(\mathcal{V}_{P,Q}(f, g) + \lambda S(g)\right) \geq \sup_{\substack{f \in \mathtt{CVX}(P) \\ f^* \in L^1(Q)}} \inf_{g \in \mathtt{CVX}(Q)} \mathcal{V}_{P,Q}(f, g),$$

which is equivalent to the inequality $J_\lambda \geq J_0$. To show that $\lim_{\lambda \to 0} J_\lambda = J_0$, it remains to prove that $\lim_{\lambda \to 0} J_\lambda \leq J_0$.

Let (f_0^*, g_0^*) be the optimal solution corresponding to J_0. Since f and g are parameterized from a convex space and $\mathcal{V}_{P,Q}(f, g)$ is a concave-convex function (as established in Lemma 6), by Theorem 5, we have:

$$\mathcal{V}_{P,Q}(f, g_0^*) \leq \mathcal{V}_{P,Q}(f_0^*, g_0^*) = J_0. \tag{11}$$

Now, consider the compactness of the function space $\mathtt{ICNN}(\mathbb{R}^d)$, there exists $(f_\lambda^*, g_\lambda^*)$ that is the optimal solution corresponding to J_λ. For this supremum-infimum formulation, we have:

$$J_\lambda = \mathcal{V}_{P,Q}(f_\lambda^*, g_\lambda^*) + \lambda S(g_\lambda^*) \leq \mathcal{V}_{P,Q}(f_\lambda^*, g) + \lambda S(g). \tag{12}$$

Note that Theorem 5 does not necessarily apply to J_λ, since $\tau(\cdot)$ may not be convex, so only one side of the saddle point inequality holds. Substituting $g = g_0^*$ into (12), we obtain:

$$J_\lambda \leq \mathcal{V}_{P,Q}(f_\lambda^*, g_0^*) + \lambda S(g_0^*).$$

Taking the limit as $\lambda \to 0$, we have:

$$\lim_{\lambda \to 0} J_\lambda \leq \lim_{\lambda \to 0} \left[\mathcal{V}_{P,Q}(f_\lambda^*, g_0^*) + \lambda S(g_0^*)\right].$$

By Lemma 4, $\lim_{\lambda \to 0} \lambda S(g) = 0$, so:

$$\lim_{\lambda \to 0} J_\lambda \leq \mathcal{V}_{P,Q}(f^*_\lambda, g^*_0). \tag{13}$$

Substituting $f = f^*_\lambda$ into (11), we have:

$$\mathcal{V}_{P,Q}(f^*_\lambda, g^*_0) \leq \mathcal{V}_{P,Q}(f^*_0, g^*_0) = J_0. \tag{14}$$

Combining (13) and (14), we obtain:

$$\lim_{\lambda \to 0} J_\lambda \leq \mathcal{V}_{P,Q}(f^*_\lambda, g^*_0) \leq \mathcal{V}_{P,Q}(f^*_0, g^*_0) = J_0.$$

Thus, with $J_\lambda \geq J_0$ and $\lim_{\lambda \to 0} J_\lambda \leq J_0$, Theorem 1 is proved. □

Proof of Theorem 2. After proving Theorem 1, we aim to show that when $\lambda \to 0$,

$$(f^*_\lambda, g^*_\lambda) \to (f^*_0, g^*_0).$$

Consider f and g from compact function spaces. This compactness is achieved by imposing arbitrary bounds on the parameters, as we discussed in Appendix C. Let $(f^*_{\lambda_n}, g^*_{\lambda_n})$ be a sequence of optimal solutions corresponding to a sequence $\lambda_n \to 0$. By the compactness of the spaces, there exists a convergent subsequence $(f^*_{\lambda_{n_k}}, g^*_{\lambda_{n_k}}) \to (f', g')$.

Since $\lim_{\lambda \to 0} J_\lambda = J_0$, we have:

$$\lim_{n \to \infty} J_{\lambda_n} = J_0.$$

Consequently,

$$\lim_{n \to \infty} J_{\lambda_n} = \lim_{n \to \infty} \mathcal{V}_{P,Q}(f^*_{\lambda_n}, g^*_{\lambda_n}) + \lambda S(g^*_{\lambda_n}) = \mathcal{V}_{P,Q}(f', g').$$

This implies that:

$$\mathcal{V}_{P,Q}(f', g') = J_0 = \mathcal{V}_{P,Q}(f^*_0, g^*_0).$$

Since $\mathcal{V}_{P,Q}(f, g) + \lambda S(g)$ is concave-convex when $\tau(\cdot)$ is convex, saddle point inequality holds for both J_λ and J_0. Note that such saddle point is unique given the objective function is strongly concave-strongly convex [21, Definition 3]. Our objective function is only concave-convex, which indicates that the convergent subsequence would possibly converge to a set of saddle points.

Therefore, we conclude that:

$$(f', g') \to (f^*_0, g^*_0).$$

□

It is important to note that when $\tau(\cdot)$ is non-convex, while a convergent subsequence may still exist, the convergence to (f^*_0, g^*_0) is not guaranteed to rigorously hold.

Appendix E. Proof of Theorem 3

We first define (still, for brevity, let $S(g) = \int_{\mathbb{R}^d} \tau(\nabla g(y) - y)\, dQ$)

$$J_{\lambda_1} = C_{P,Q} + \sup_{\substack{f \in \mathtt{CVX}(P) \\ f^* \in L^1(Q)}} \inf_{g \in \mathtt{CVX}(Q)} \mathcal{V}_{P,Q}(f, g) + \lambda_1 S(g),$$

$$J_{\lambda_2} = C_{P,Q} + \sup_{\substack{f \in \mathtt{CVX}(P) \\ f^* \in L^1(Q)}} \inf_{g \in \mathtt{CVX}(Q)} \mathcal{V}_{P,Q}(f, g) + \lambda_2 S(g).$$

Consider $(f^*_{\lambda_1}, g^*_{\lambda_1})$ and $(f^*_{\lambda_2}, g^*_{\lambda_2})$ be the optimal solution to respect J_{λ_1} and J_{λ_2}, we aim to prove that given $\lambda_1 < \lambda_2$, we have

$$S(g^*_{\lambda_1}) \geq S(g^*_{\lambda_2}).$$

By the saddle point inequality, we have the following four inequalities,

$$\mathcal{V}_{P,Q}(f^*_{\lambda_1}, g^*_{\lambda_1}) + \lambda_1 S(g^*_{\lambda_1}) \leq \mathcal{V}_{P,Q}(f^*_{\lambda_1}, g^*_{\lambda_2}) + \lambda_1 S(g^*_{\lambda_2}), \tag{15}$$

$$\mathcal{V}_{P,Q}(f^*_{\lambda_2}, g^*_{\lambda_2}) + \lambda_2 S(g^*_{\lambda_2}) \leq \mathcal{V}_{P,Q}(f^*_{\lambda_2}, g^*_{\lambda_1}) + \lambda_2 S(g^*_{\lambda_1}), \tag{16}$$

$$\mathcal{V}_{P,Q}(f^*_{\lambda_1}, g^*_{\lambda_1}) + \lambda_1 S(g^*_{\lambda_1}) \geq \mathcal{V}_{P,Q}(f^*_{\lambda_2}, g^*_{\lambda_1}) + \lambda_1 S(g^*_{\lambda_1}), \tag{17}$$

$$\mathcal{V}_{P,Q}(f^*_{\lambda_2}, g^*_{\lambda_2}) + \lambda_2 S(g^*_{\lambda_2}) \geq \mathcal{V}_{P,Q}(f^*_{\lambda_1}, g^*_{\lambda_2}) + \lambda_2 S(g^*_{\lambda_2}). \tag{18}$$

By (17) and (18), we have

$$\mathcal{V}_{P,Q}(f^*_{\lambda_1}, g^*_{\lambda_1}) \geq \mathcal{V}_{P,Q}(f^*_{\lambda_2}, g^*_{\lambda_1}), \tag{19}$$

$$\mathcal{V}_{P,Q}(f^*_{\lambda_2}, g^*_{\lambda_2}) \geq \mathcal{V}_{P,Q}(f^*_{\lambda_1}, g^*_{\lambda_2}). \tag{20}$$

Rearranging (15) and (16), we have

$$\mathcal{V}_{P,Q}(f^*_{\lambda_1}, g^*_{\lambda_1}) - \mathcal{V}_{P,Q}(f^*_{\lambda_1}, g^*_{\lambda_2}) + \mathcal{V}_{P,Q}(f^*_{\lambda_2}, g^*_{\lambda_2}) - \mathcal{V}_{P,Q}(f^*_{\lambda_2}, g^*_{\lambda_1}) \leq (\lambda_1 - \lambda_2)[S(g^*_{\lambda_2}) - S(g^*_{\lambda_1})]$$

With (19) and (20), we can get

$$0 \leq (\lambda_1 - \lambda_2)[S(g^*_{\lambda_2}) - S(g^*_{\lambda_1}),]$$
$$S(g^*_{\lambda_1}) \geq S(g^*_{\lambda_2}), \quad (\because \lambda_1 < \lambda_2)$$

□

Appendix F. Sparsity-Inducing Penalties τ

ℓ_1 and ℓ_0-Norms. We denote ℓ_1 and ℓ_0 by the norms $\|\cdot\|_1$ and $\|\cdot\|_0$, respectively. The ℓ_1-norm directly penalizes the magnitude of the displacement vector, encouraging it to move towards the nearest target and inducing sparsity. In contrast, the ℓ_0-norm penalizes the dimensionality of the displacement vector by counting the number of non-zero components. However, we empirically find that the ℓ_0-norm does not effectively induce sparsity in the map learned via ICNN.

Vanishing Shrinkage STVS. Schreck et al. [17] introduce the soft-thresholding operator with vanishing shrinkage (STVS) for displacement vector $\mathbf{z}$,

$$\tau_{\text{stvs}}(\mathbf{z}) = \gamma^2 \mathbf{1}_d^T \left(\sigma(\mathbf{z}) + \frac{1}{2} - \frac{1}{2} e^{-2\sigma(\mathbf{z})} \right),$$

where $\sigma(\mathbf{z}) = \operatorname{asinh}(\frac{z}{2\gamma})$ with element-wise operations. Cuturi et al. [9] add $+\frac{1}{2}$ to make τ_{stvs} non-negative.

***k*-overlap Norm.** Argyriou et al. [2] define the k-overlap norm for any vector $\mathbf{z} \in \mathbb{R}^d$ as:

$$\|\mathbf{z}\|_{ovk} := \min \left\{ \sum_{I \in \mathcal{G}_k} \|v_I\|_2 : \text{supp}(v_I) \subseteq I, \sum_{I \in \mathcal{G}_k} v_I = \mathbf{z} \right\},$$

where $\mathcal{G}_k$ represents the set of all subsets of $\{1, \ldots, d\}$ with cardinality at most k. For any vector $\mathbf{z} \in \mathbb{R}^d$, let $\mathbf{z}^\downarrow$ denote the vector formed by arranging all entries of $\mathbf{z}$ in decreasing order. This expression can be evaluated as follows to represent an ℓ_1/ℓ_2 norm split among the d variables in the vector:

$$\|\mathbf{z}\|_{ovk}^2 = \sum_{i=1}^{k-r-1} (|\mathbf{z}_i^\downarrow|)^2 + \left(\sum_{i=k-r}^{d} |\mathbf{z}_i^\downarrow| \right)^2 / (r+1),$$

where $r \leq k-1$ is the unique integer such that

$$|\mathbf{z}_{k-r}^\downarrow| \leq \sum_{i=k-r}^{d} |\mathbf{z}_i^\downarrow| < |\mathbf{z}_{k-r-1}^\downarrow|.$$

Smoothed ℓ_0-Norm. Mohimani et al. [14] propose a continuous approximation to the ℓ_0-norm by replacing the discontinuous function $\nu(z)$, defined as:

$$\nu(z) = \begin{cases} 1 & z \neq 0 \\ 0 & z = 0 \end{cases},$$

with a smooth Gaussian function:

$$f_\sigma(z) = \exp\left(-\frac{z^2}{2\sigma^2} \right). \tag{21}$$

As $\sigma \to 0$, $f_\sigma(z)$ approximates $1 - \nu(z)$. The smoothed approximation of the ℓ_0-norm is given by:

$$n - F_\sigma(\mathbf{z}), \quad \text{where} \quad F_\sigma(\mathbf{z}) = \sum_{i=1}^{n} f_\sigma(z_i).$$

Appendix G. Convergence Analysis for Low Dimensional Experiments

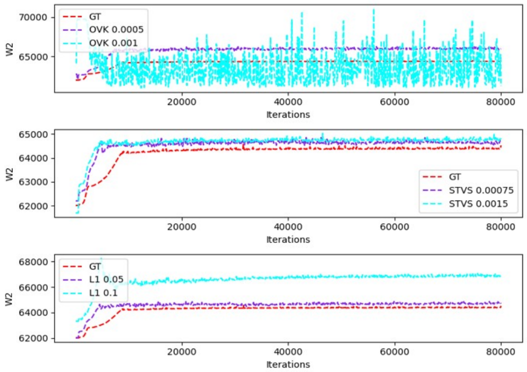

Figure 5: Convergence results of the Wasserstein distance in Figure 1. 'GT' represents the unbiased ground truth Wasserstein distance, as learned from equation (5). The other lines represent biased Wasserstein distances obtained using different penalty functions, learned from equation (6). For example, 'OVK 0.0005' indicates the use of a k-overlap penalty with $\lambda = 0.0005$.

We first construct an "8-Gaussian" example, where the source is located at the center, and the target points are generated using eight Gaussian distributions. Unlike the "8-Gaussian" examples in ICNN OT [13], we make the range of the Gaussians more compact to test how ICNN could handle this situation.

Among the three penalties we use, the k-overlap norm has the issue of failing to converge and does not produce a desirable sparse map. Its quadratic nature makes it highly sensitive to the intensity parameters. As λ increases, the k-overlap norm tends to 'over-transport' — moving the source even further towards the target than intended. Therefore, we recommend applying the ℓ_1-norm and τ_{stvs} to the ICNN, as their convergence trends align with our expectations from Theorem 1.

It is important to note that the Wasserstein distance typically converges quickly in the initial stages. However, ICNN usually requires additional time to make minor adjustments to the transportation map. Leveraging this characteristic, we developed our heuristic framework to dynamically adjust the sparsity intensity during the training process.

We also make an additional dataset "2-Box" for reference, as shown in Figure 6:

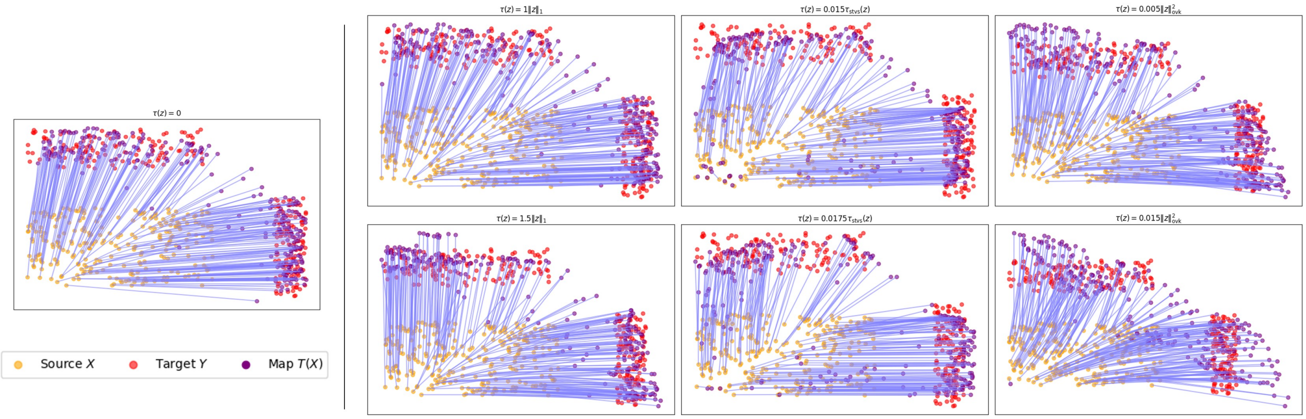

Figure 6: Targets are randomly generated from two boxes positioned next to the source. OT maps are learned using different penalty functions and intensity parameters λ between the source and target. We expect the displacement vectors to be either horizontal or vertical, allowing for a clearer interpretation of the transportation trend.

Appendix H. Simulated Annealing Framework for λ Adjustment (Low Dimensional Space)

We embed the simulated annealing framework into the ICNN training process to optimize the sparsity-inducing intensity λ. Initially, we perform n_{ini} iterations using the starting value of λ to train the functions f and g until the map's sparsity and error levels converge to a stable state.

After reaching this initial stability, we initiate the simulated annealing process with a starting temperature T. During this phase, λ is adjusted randomly within a range determined by the current temperature; a higher T allows for a wider range of adjustments. We then continue training f and g with the new λ for n_{tr} iterations. We observe that ICNN can quickly adapt to changes in λ.

Once the map's sparsity and error levels stabilize again, we evaluate whether to accept the new λ based on its associated evaluation metric. To ensure smooth transitioning, if the new λ is not accepted, we perform an additional n_{sm} iterations of training to allow the ICNN to revert smoothly to the previous value of λ.

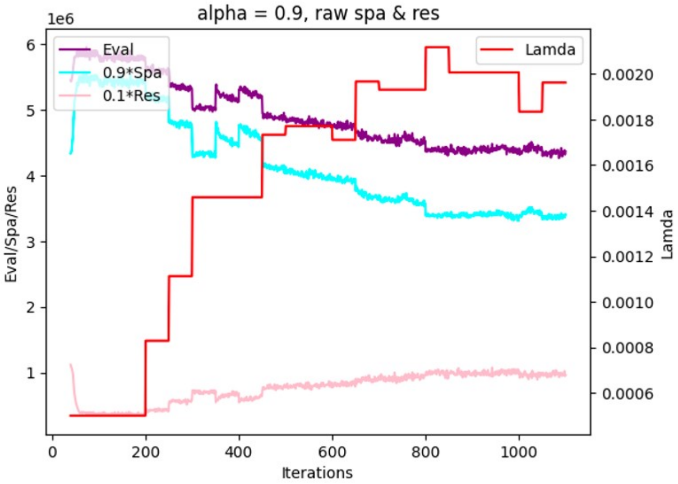

Figure 7: Simulation of Algorithm 1 for Figure 2 with $\alpha = 0.9$. All parameters in the figure are plotted every 100 iterations. The simulated annealing process concludes after approximately 120,000 iterations of training.

This algorithm is illustrated in Algorithm 1. Figure 7 shows an example simulation of Algorithm 1. Note that all the hyperparameters related to iterations need to be re-adjusted based on the size of the data.

Hyperparameters. We use most of the default hyperparameters for ICNN training. For iterations related hyperparameters, we set $n_{ini} = 20,000$ and $n_{tr} = n_{sm} = 2,000$. For simulated annealing hyperparameters, we set initial temperature as 1.0, and minimum temperature as 0.15, temperature decay rate as 0.9, and range adjustment parameter $p = 3$. For τ_{stvs} penalty, we set $\gamma = 100$.

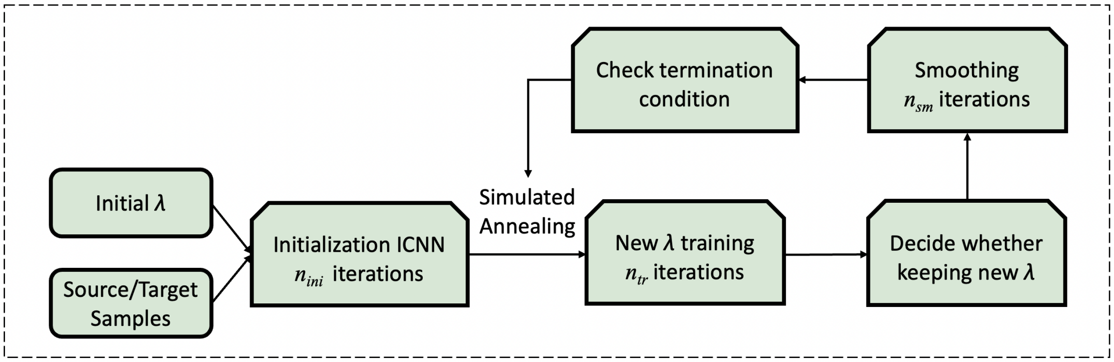
Check termination condition
Smoothing n_{sm} iterations
Initial λ
Source/Target Samples
Initialization ICNN n_{ini} iterations
Simulated Annealing
New λ training n_{tr} iterations
Decide whether keeping new λ

Figure 8: Training structure of SICNN, with the simulated annealing module embedded into the original ICNN OT. Specific details are shown in Algorithm 1 below.

Algorithm 1: Simulated Annealing with SICNN (Low Dimensional Setting)

```
Input: Initialization iterations $n_{\text{ini}}$, Training iterations $n_{\text{tr}}$, Smoothing iterations $n_{\text{sm}}$
Input: Initial temperature $T$, Minimum temperature $T_{\min}$, Temperature decay rate $d$, Range
       adjusting parameter $p$
Input: Sparsity-feasibility trade-off parameter $\alpha$, Initial sparsity-inducing intensity $\lambda$
for $i = 1$ to $n_{ini}$ do
    Map $\leftarrow$ Learn $f$ and $g$ with $\lambda$;
end
Eval $\leftarrow \alpha \cdot \text{Spa(Map)} + (1 - \alpha) \cdot \text{Err(Map)}$;
while $T > T_{\min}$ do
    Range $\leftarrow \max(0.01, \exp(-p \cdot (1 - T)))$;
    $\lambda_{\text{new}} \leftarrow \lambda \cdot (1 + \text{random}(-\text{Range}, \text{Range}))$;
    for $j = 1$ to $n_{tr}$ do
        Map $\leftarrow$ Learn $f$ and $g$ with $\lambda_{\text{new}}$;
    end
    $\text{Eval}_{\text{new}} \leftarrow \alpha \cdot \text{Spa(Map)} + (1 - \alpha) \cdot \text{Err(Map)}$;
    $\Delta \leftarrow \text{Eval}_{\text{new}} - \text{Eval}$;
    if $\Delta > 0$ or random() $< \exp(\Delta / T)$ then
        $\lambda \leftarrow \lambda_{\text{new}}$;
    else
        Do nothing;
    end
    for $k = 1$ to $n_{sm}$ do
        Map $\leftarrow$ Learn $f$ and $g$ with $\lambda$;
    end
    $T \leftarrow T \cdot d$;
end
```

Appendix I. Theoretical Solution towards formulation (8) - Regularized Gradient Descent Method (RGD)

Instead of directly solving formulation (8) with its constraints, we approach this problem using an unregularized ground truth solution T_{GT}, defined as:

$$T_{GT} = \underset{T:T_{\#}P=Q}{\arg\min} \frac{1}{2}\mathbb{E}_{X\sim P}\|T(X) - X\|^2.$$

We convert the constraint in (8) into a regularization term. For any displacement vector $\mathbf{z}$, we define:

$$\tau(\mathbf{z}) = \max(0, \text{smoothed } \|\mathbf{z}\|_0 - l),$$

where the penalty ceases once the displacement vector meets the dimensionality requirement l. Thus, the problem is reformulated as a multi-objective optimization task that can be solved via gradient descent, starting from the initial solution T_{GT}:

$$\min_T \beta D(T(X), Y) + \mu \sum_{x\in X} \tau(T(x) - x). \tag{22}$$

To improve computational efficiency, we approximate $D(T(X), Y)$ by $\|T(X) - T_{GT}(X)\|^2$, leveraging the fact that T_{GT} satisfies $T_{\#}P = Q$. The core objective is to ensure that the displacement vectors have low dimensionality while not deviating significantly from the ground truth. The parameters β and μ can be adjusted dynamically based on the feedback from the gradient descent solution: increasing μ if the dimensionality constraint is not satisfied, and decreasing it when the constraint is met to enhance solution feasibility.

We perform gradient descent to update the displacement vectors using the following update rule:

$$T(x_i)^{(k+1)} = T(x_i)^{(k)} - r \cdot \nabla_{T(x_i)} J(T),$$

where r is the learning rate, and $\nabla_{T(x_i)} J(T)$ is the gradient of the objective function (22) with respect to the vector $T(x_i)$.

The main drawback of this method is the requirement to compute an unregularized map before performing gradient descent. Furthermore, the displacement vectors derived from the gradient descent process no longer exhibit the typical characteristics of an optimal transport map or plan. As a result, their interpretation is restricted to the specific sample and cannot be generalized or applied as a map or plan to other samples.

Appendix J. Heuristic Framework for λ Adjustment (High Dimensional Space)

Our method is straightforward: we first increase λ to ensure that the displacement vectors meet the dimensionality constraint. Then, we heuristically decrease λ to reduce the intensity of sparsity-inducing, allowing the model to learn a more feasible map. The details of this algorithm are illustrated in Algorithm 2.

Algorithm 2: Simulated Annealing with SICNN (High Dimensional Setting)

```
Input: Initialization iterations $n_{\text{ini}}$, Training iterations $n_{\text{tr}}$, Smoothing iterations $n_{\text{sm}}$
Input: Initial temperature $T$, Minimum temperature $T_{\min}$, Temperature decay rate $d$, Range
       adjusting parameter $p$
Input: Initial sparsity-inducing intensity $\lambda$, Intensity increasing rate $ir$, Dimension
       requirement $l$
for $i = 1$ to $n_{ini}$ do
    Map ← Learn $f$ and $g$ with $\lambda$ using ICNN;
end
Dim ← Dim(Map);
while $Dim \geq l$ do
    $\lambda \leftarrow \lambda \cdot (1 + ir)$;
    for $j = 1$ to $n_{tr}$ do
        Map ← Learn $f$ and $g$ with $\lambda$ using ICNN;
    end
    Dim ← Dim(Map);
end
while $T > T_{\min}$ do
    Range ← $\max(0.01, \exp(-p \cdot (1 - T)))$;
    $\lambda_{\text{new}} \leftarrow \lambda \cdot (1 + \text{random}(-\text{Range}, 0))$;
    for $j = 1$ to $n_{tr}$ do
        Map ← Learn $f$ and $g$ with $\lambda_{\text{new}}$ using ICNN;
    end
    Dim_new ← Dim(Map);
    if $Dim_{new} \geq l$ then
        $\lambda \leftarrow \lambda_{\text{new}}$;
    end
    for $k = 1$ to $n_{sm}$ do
        Map ← Learn $f$ and $g$ with $\lambda$ using ICNN;
    end
    $T \leftarrow T \cdot d$;
end
```

Appendix K. High Dimensional Experiment Setup and Results

In our experiment, we use 500 cells, each characterized by 250 genes (i.e., in $\mathbb{R}^{250}$). Among these genes, we randomly select 20 to undergo a significant perturbation in their expression levels (with an intensity of 100) and add random noise to the other genes to simulate changes in gene expression under drug stimulation. We learn an OT mapping between the cells in the original group (before drug perturbation) and the treated group (after drug perturbation).

Our objective is to reduce the dimensionality of the displacement vectors to around 20 ($l = 21$ due to the precision problem of ICNN), capturing only the genes affected by the drug while minimizing the influence of noise from unaffected genes. We compares the result of four different methods to learn this mapping: unregularized ground truth (`py.emd`), regularized gradient descent method (RGD, Appendix I), ICNN OT [13], and SICNN. Example of displacement vector is shown below:

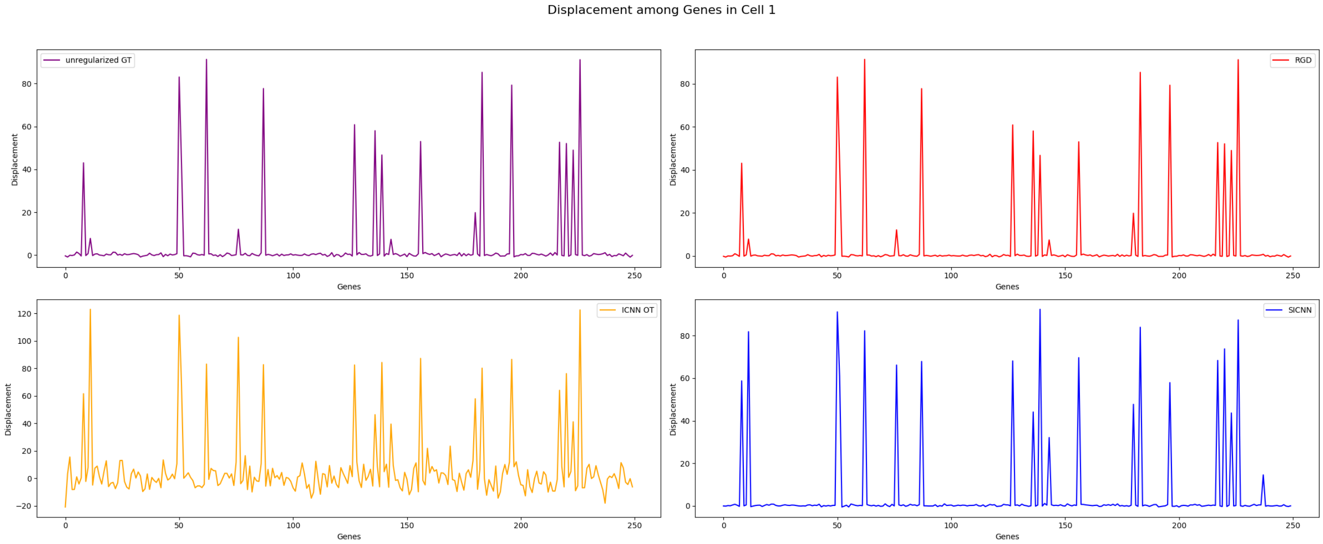

Figure 9: The displacement mapping of cell 1 from the original group to the treated group across four different methods.

Part of the numerical results for Figure 4 is presented in Table 1. Among the noise gene categories, we observe a significant dimensionality reduction of displacement vectors from the ICNN OT solution to the SICNN solution. In contrast, even though the displacement entries for noise genes are reduced, the displacement of perturbed genes produced by SICNN still maintain the similar numerical level to that of the unregularized ground truth (GT) and RGD.

For RGD and the unregularized GT, we find that RGD successfully reduces the numerical values of the noise genes while preserving the numerical levels of the perturbed genes, keeping them comparable to those in the unregularized GT.

For the final average dimension of all displacement vectors, the unregularized GT and ICNN OT methods yield average dimensions of 31 and 208.5, respectively, whereas the RGD and SICNN methods produce significantly lower average dimensions of 20 and 20.88, respectively.

Table 1: Example of Displacement Vector Entries for Cell 1 Across Different Methods

Gene	**Method**			
	Unregularized GT	**RGD**	**ICNN OT**	**SICNN**
Noise gene				
1	-0.3462	-0.2054	-20.8935	-0.0943
2	-0.8874	-0.5546	3.1371	-0.1606
3	-0.0587	-0.0346	15.5107	0.0882
4	-0.1844	-0.1087	-8.1908	-0.004
5	0.1065	0.0627	-8.1025	0.3725
6	1.4298	1.0052	1.1131	0.7004
7	0.767	0.4716	-4.1726	0.2827
8	-0.4	-0.238	0.3409	-0.2775
10	-0.2009	-0.1185	-2.2892	-0.0279
11	0.7323	0.4485	7.8398	0.3889
13	-0.2199	-0.1298	-4.9439	-0.4263
14	0.4978	0.2982	7.3894	-0.1368
15	0.635	0.3848	8.883	0.1079
16	0.0332	0.0195	1.1512	0.2405
17	-0.1008	-0.0593	-4.2023	0.2815
18	-0.309	-0.183	4.9597	-0.3348
19	0.4995	0.2992	12.7438	0.1468
20	0.1725	0.1017	-6.0698	0.5856
21	0.1528	0.09	-3.5468	0.2549
22	1.3816	0.9578	-2.9502	0.7722
23	1.2688	0.8537	-7.5327	0.7564
24	0.0435	0.0256	-3.2566	0.1695
Perturbed gene				
9	43.0514	43.0514	61.5517	58.7538
12	7.7989	7.7989	122.9864	81.8804
51	83.0593	83.0593	118.6831	91.2122
52	43.7719	43.7719	69.5696	60.9674
63	91.2791	91.2791	83.1085	82.323
77	12.0843	12.0843	102.573	66.1896
88	77.6835	77.6835	82.7079	67.8737
128	60.8325	60.8325	82.4737	68.1296
137	58.0109	58.0109	46.2925	44.1126
140	46.694	46.694	84.2862	92.4234
157	52.9928	52.9928	87.2424	69.6636
184	85.2533	85.2533	80.1477	83.9621
197	79.2744	79.2744	86.5302	57.9
Overall Dimension (Noise + Perturbed gene)	31	20	115	21

Hyperparameters for the RGD Method. To ensure reproducibility, we provide the details of the parameters used in our high-dimensional experiments. The parameter for the smoothed ℓ_0-norm in Equation (21) is set to $\sigma = 1$. For the multi-objective optimization, we use $\beta = 1$ and $\mu = 1.4$. The gradient descent employs a learning rate of $r = 0.01$ and a convergence threshold of 10^{-6}.

Hyperparameters for the ICNN OT Method. We largely used the default hyperparameter settings as outlined in ICNN OT [13], with a modification to the batch size, which was set to 128 to better accommodate our datasets.

Hyperparameters for SICNN method. We set $n_{ini} = 10,000$ and $n_{tr} = n_{sm} = 5,000$. For simulated annealing hyperparameters, we set initial temperature as 1.0, and minimum temperature as 0.15, temperature decay rate as 0.9, and range adjustment parameter $p = 3$. The parameter for the smoothed ℓ_0-norm in Equation (21) is set to $\sigma = 1.0$. Initial $\lambda = 0.005$, intensity increasing rate $ir = 0.5$, and dimension requirement $l = 21$.

Appendix L. Conclusions

In this work, we propose an intuitive approach to incorporate displacement sparsity into ICNN to solve OT problems. Unlike the constant sparsity intensity used in Cuturi et al.'s [9] displacement-sparse map, we introduce a novel heuristic framework that dynamically adjusts the sparsity intensity throughout ICNN training based on different objective functions. Furthermore, we apply our heuristic framework to solve the direct displacement-constrained problem in high-dimensional settings.

Future Work. In this short version of SICNN, we regret that we have only presented experimental results on synthesized datasets. However, we have already retrieved multiple datasets for 2D cell perturbation and high-dimensional gene perturbation tasks. In future work, we will conduct additional experiments to evaluate the performance of SICNN on these real-world datasets. We will also develop a more effective method for setting the training parameters, such as n_{sm} and n_{tr}, in the heuristic framework. Currently, these parameters are conservatively chosen in an arbitrary manner. Implementing an improved mechanism to determine these parameters could significantly accelerate the training process.