Embedding Empirical Distributions for Computing Optimal Transport Maps

Mingchen Jiang[2*] Peng Xu[3,4*†] Xichen Ye[1] Xiaohui Chen[4] Yun Yang[5] Yifan Chen[1†]
[1] Hong Kong Baptist University [2] Institute of Science Tokyo [3] University of Illinois Urbana-Champaign
[4] University of Southern California [5] University of Maryland, College Park
* Equal contribution. This work was performed while the first author was interning at Hong Kong Baptist University.
† Correspondence to: Peng Xu ⟨pengxu1@illinois.edu⟩, Yifan Chen ⟨yifanc@hkbu.edu.hk⟩.

Abstract

Distributional data have become increasingly prominent in modern signal processing, highlighting the necessity of computing optimal transport (OT) maps across *multiple* probability distributions. Nevertheless, recent studies on neural OT methods predominantly focused on the efficient computation of a *single* map between two distributions. To address this challenge, we introduce a novel approach to learning transport maps for new empirical distributions. Specifically, we employ the transformer architecture to produce embeddings from distributional data of varying length; these embeddings are then fed into a hypernetwork to generate neural OT maps. Various numerical experiments were conducted to validate the embeddings and the generated OT maps. The model implementation and the code are provided on https://github.com/jiangmingchen/HOTET.

I. Introduction

Optimal transport (OT) theory [1] is a mathematical framework for finding the most efficient way (in the sense of minimizing a given cost function) to transport one probability distribution to another. When the quadratic cost is used, OT theory induces a metric space for probability measures, and the distance thereof is referred to as the 2-Wasserstein metric [2]. This notion provides a geometric view of distributions, and therefore makes OT an invaluable tool in information theory [3]–[6]. Furthermore, OT has already been used in many applications, such as flow-based diffusion models [7], [8], GANs [9], [10], style transfer [11], data embedding [12], [13], multilingual alignment [14], [15], domain adaptation [16], [17], and model compression [18]–[20].

Challenges. In many applications, transport maps between n source distributions and a single target distribution are desired. For instance, in the Wasserstein embedding [12] scheme, input distributions are represented by the OT maps that link them to a reference distribution. Another example is the color transfer technique, where the color histogram of a reference image is transformed to match that of other images. To obtain OT maps in these settings, conventional approaches require the use of $2n$ neural networks to model the dual variables in the OT problem. If a new OT map is needed between a different source measure and the target measure, it must be computed from scratch. Computation of a single OT map is already challenging, and the difficulty increases considerably when seeking transport maps between numerous new source distributions and a single target distribution.

Overview. To make the computation of multiple OT maps more efficient and generalizable, we propose a new paradigm (illustrated in the right panel of Figure 1) to learn the OT maps between multiple source distributions $\mu_1, \ldots, \mu_n$ and a single target distribution ν. In short, samples from the source distributions are passed to a transformer-based module $\mathcal{E}$ to produce embeddings in $\mathbb{R}^d$. These embeddings are then feed into hypernetworks $\mathcal{F}$ and $\mathcal{G}$ to generate parameters for the potential networks, whose gradients approximate the OT maps. During training, samples from ν are passed to the potential networks for loss computation, which propagates ν's information into $\mathcal{F}$ and $\mathcal{G}$. Notably, no explicit embedding of ν is required after the training is complete.

We refer to the whole framework as the Hypernetworks for Optimal Transport with Embedding Transformers (**HOTET**). One notable strength of HOTET is that the embeddings of the input distributions in $\mathbb{R}^d$ are directly obtained, which can then be used in downstream learning tasks.

A. Related Works

There are some works that use transformers to deal with OT problems, such as embedding empirical distributions into a latent space wherein Euclidean distances approximate OT distances [21], regularizing the training process [22], and neural network fusion [23]. And our focus is on generating transport maps without engaging in direct computation by using transformers.

Generally speaking, the idea of generating transport maps without direct computation is not new, as it is closely related to distributional regression [24], [25]. Recent computational developments along this line include CondOT [26], Meta OT [27], and GeONet [28].

* Equal contribution.

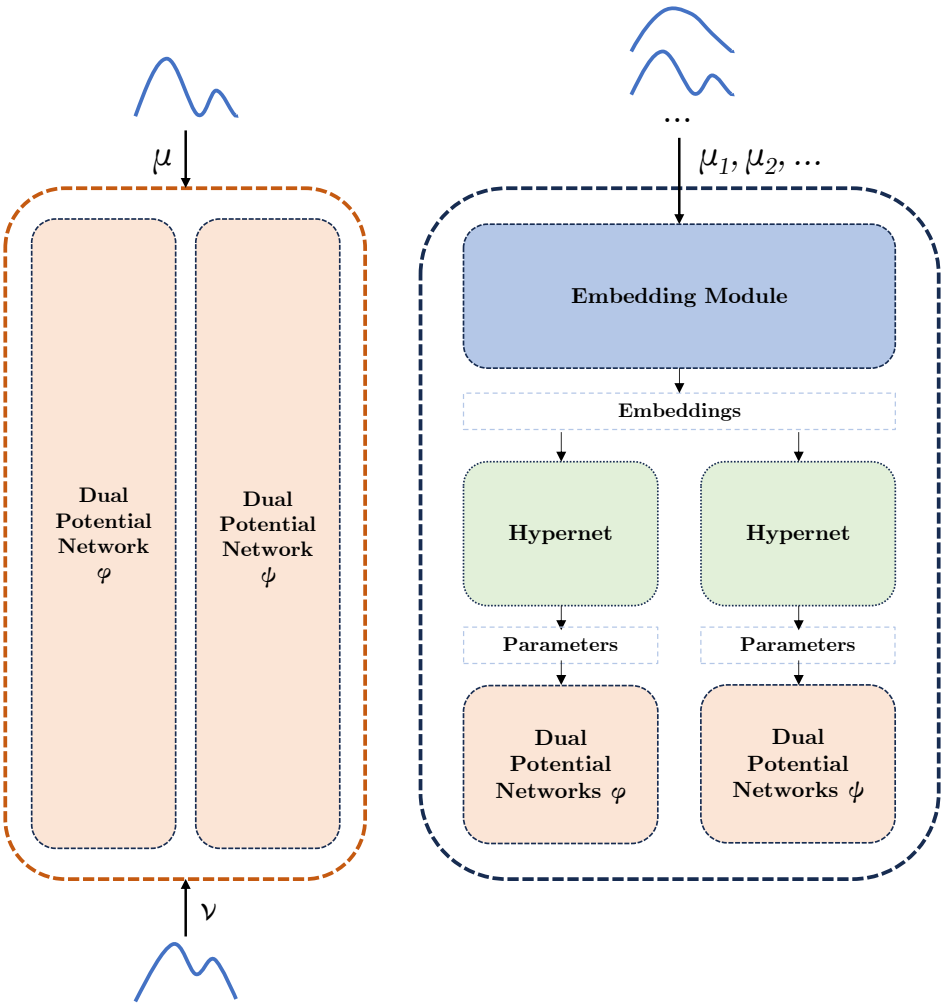


Fig. 1: The network architectures for direct computation of OT maps (left), and for our proposed method of generating OT maps through hypernetworks (right).

In more detail, CONDOT [26] proposed to estimate a family of OT maps conditioned on a context variable, which can then be generalized given new context; in Meta OT [27], amortized optimization was used to predict OT maps from the input measure. Unlike the previous two, GeONet [28] learned neural operators to generate the *Wasserstein geodesics* connecting the pair of input measures, which can be regarded as an extension to the aforementioned methods in the context of dynamic OT problem. However, this method falls out of the scope of neural OT learning, the focus of our project.

We defer a more comprehensive comparison with existing methods to Appendix B-E.

B. Our Contributions

We summarize the contributions of this work as follows:

- We proposed a new paradigm for learning neural OT maps and distribution embeddings for multiple distributions.
- We employed existing benchmarks [29] and conducted various tasks to demonstrate the effectiveness of our design.

II. PRELIMINARIES

We start the review with the necessary notations for OT in Section II-A. We then introduce ICNN, hypernetwork, and Transformer, in Section II-B. Moreover, a more comprehensive review, and the connection between attention and kernel estimators, are provided in Appendix B; building on these we propose to embed an empirical distribution (observation points with arbitrary sample weights) through a transformer in Section III-B, so as to generate embeddings for hypernetworks. Notations in this work are collected in Appendix C-A.

A. Optimal Transport

Optimal transport, as the name suggests, is an optimization problem. We will review its properties in this subsection.

Monge's problem. Let $\mathcal{P}_2(\mathbb{R}^d)$ denote the set of probability measures over $\mathbb{R}^d$ with finite second moments. Given $\mu, \nu \in \mathcal{P}_2(\mathbb{R}^d)$, Monge seeks a map T that push forwards μ to ν while minimizing the transportation cost. With cost $c(x,y) = \|x-y\|_2^2$, a metric W_2 on the space $\mathcal{P}_2(\Omega)$ is induced as

$$W_2^2(\mu,\nu) \coloneqq \inf_{T_{\#}\mu=\nu} \int_{\mathbb{R}^d} \|T(x)-x\|_2^2 \,\mathrm{d}\mu(x), \tag{MP}$$

where $T_{\#}\mu = \nu$ means that $\nu(A) = \mu\big(T^{-1}(A)\big)$ for every Borel-measurable set $A \subset \mathbb{R}^d$.

Kantorovich's relaxation. The constraint in (MP) is highly nonlinear, which makes the solution difficult to obtain. Thus, a linear programming relaxation of (MP), introduced by Kantorovich, is more commonly used in computation:

$$W_2^2(\mu,\nu) \coloneqq \inf_{\gamma\in\Pi(\mu,\nu)} \int_{\mathbb{R}^d\times\mathbb{R}^d} \|x-y\|_2^2 \,\mathrm{d}\gamma(x,y). \tag{KR}$$

The constraint set $\Pi(\mu,\nu)$ in (KR) is the set of couplings between μ and ν, i.e., probability measures $\gamma \in \mathcal{P}(\mathbb{R}^d \times \mathbb{R}^d)$ that satisfy $(\pi_x)_{\#}\gamma = \mu$ and $(\pi_y)_{\#}\gamma = \nu$, where $\pi_x(x,y) = x$, $\pi_y(x,y) = y$ for all $(x,y) \in \mathbb{R}^d \times \mathbb{R}^d$.

Dual Formulation. The relaxation (KR) admits a dual problem [1, Thm. 1.3], which reads

$$\begin{aligned} \frac{1}{2}W_2^2(\mu,\nu) = \sup_{\varphi,\psi} \quad & \left\{ \int_{\mathbb{R}^d} \varphi \,\mathrm{d}\mu + \int_{\mathbb{R}^d} \psi \mathrm{d}\nu \right\} \\ \text{s.t.} \quad & (\varphi,\psi) \in L^1_\mu \times L^1_\nu \\ & \varphi(x) + \psi(y) \le \frac{1}{2}\|x-y\|_2^2 \end{aligned} \tag{DP}$$

The optimal solution for the dual exists [30, Prop. 1.11] and is known as *Kantorovich potentials*.

The formulation (DP) and its extensions are adopted in multiple computational methods for their convenience. The conjugacy in the dual variables is typically enforced through regularization, cf. [29] for a comprehensive review. Once the optimal dual potentials φ^*, ψ^* are obtained, the OT map (whenever exists) can be recovered as $T^*(x) = \nabla\tilde{\varphi}(x)$, where $\tilde{\varphi}(x) \coloneqq \|x\|_2^2/2 - \varphi^*(x)$ is a convex function. It follows that

$$(T^*)^{-1}(y) = y - \nabla\psi^*(y).$$

In other words, the OT map and its inverse can be obtained by differentiating the solutions of (DP).

B. Neural Architectures

We then introduce the neural components in learning.

Input Convex Neural Networks [31, ICNN] are utilized for modeling the convex potential function $\tilde{\varphi}(\cdot)$ in many neural OT implementations [29]. More details are deferred to Appendix B-A.

Hypernetwork [32] is another component in our paradigm. Broadly speaking, hypernetwork refers to the neural network architecture designed to generate parameters for another neural network. Given a target network f_θ, instead of directly learning parameters θ from data as traditional learning, hypernetworks generate θ as an output mapped from a certain context variable. This allows for more flexible and efficient learning, particularly in tasks with complex or varying structures. Hypernetworks have been used in tasks such as neural architecture search, meta-learning, and conditional generation [33].

Transformers [34], equipped with the attention modules, are primarily used in natural language processing (NLP) tasks. The attention modules in transformers follow the spirit of nonparametric methods and can address sequences of indefinite length. After removing the positional encoding for addressing sequences, transformers are proven universal approximators for set-to-set maps [35, Theorem 2], thus appropriate to handle empirical distributional data (a set of samples). A complete introduction to transformers are deferred to Appendix B-B.

While the complexity of attention is quadratic, efficient GPU implementations, such as FlashAttention [36], [37], have dramatically accelerated the computation while maintaining a linear space complexity. Our proposed method has incorporated the open-shelf efficient implementations as well.

III. Hypernetworks for Optimal Transport with Embedding Transformers

In this section, we discuss several crucial aspects of HOTET, our proposed training paradigm. Due to space limit, implementation details are deferred to Appendix A-A.

TABLE I: Performance of the constructed forward (fwd) and inverse (inv) transport maps ($\mathcal{L}^2$-UVP (%) as the metric). Lower implies the fitted map approximates the true OT map better. The standard deviation is calculated over 10 runs.

DIM	HOTET (fwd)	MetaOT (fwd)	MM-B (fwd)	HOTET (inv)	MetaOT (inv)	MM-B (inv)
2	5.03 ±0.33	11.71 ±0.49	0.54 ±0.06	0.66 ±0.03	15.52 ±0.56	0.81 ±0.03
4	10.06 ±0.39	11.63 ±0.46	1.91 ±0.10	2.01 ±0.04	9.27 ±0.42	0.48 ±0.07
8	11.38 ±0.41	23.18 ±0.42	5.51 ±0.15	3.80 ±0.23	20.50 ±0.87	2.22 ±0.05
16	16.91 ±0.58	47.61 ±1.03	13.17 ±0.08	4.64 ±0.18	34.16 ±0.47	6.27 ±0.32
32	20.87 ±0.56	59.99 ±0.67	25.87 ±0.59	17.03 ±0.47	44.60 ±0.78	10.20 ±0.26
64	37.62 ±0.84	74.02 ±0.80	23.63 ±0.25	15.07 ±0.37	30.88 ±0.89	13.95 ±0.52

A. Base OT solvers in HOTET

Training the hypernetworks in HOTET requires a base OT solver to learn to generate neural networks that approximate true OT maps (though in principle the HOTET framework is agnostic to the choice of the base solver, provided it delivers accurate approximations). Mainstream OT solvers are mainly maximization-minimization-based, which are subject to divergence in training; for the numerical experiments in this work, we selected the base solver in use from two instances for numerical stability, referred to as MM-B and MMv2 in [29]. Both solvers utilize the dual OT formulation and ICNNs to approach the convex potentials φ and ψ, whose gradients serve as approximations of the OT maps. Usage of the solvers is discussed in Section IV-A, and detailed formulations / implementations for these solvers are provided in Appendix A-B.

B. Embedding empirical distributions

The key step in our training paradigm is to generate context embeddings from the input **empirical distributions**. We employ transformers in this task for several reasons:

1) With the positional encoding removed, transformers are universal approximators for set-to-set maps [35, Thm. 2] and permutation invariant functions [38, Prop. 2], which match the characteristics of an empirical distribution.
2) Transformers are suitable for inputs with variable sizes, which is not well handled by previous methods [26], [27].
3) The architecture of the transformer allows it to take the weights from the empirical distribution into consideration. This aspect is explored in Appendices B-C and B-D.

A simplified process for HOTET to embed empirical distributions can be described as follows: a design matrix $\boldsymbol{X}$ (the input empirical distribution) is passed to the L blocks in our transformer embedding module; the first $L-1$ blocks follow a regular design in [34], which keeps the input sample dimension hd (h is the number of heads, see Appendix B-B); the MLP sub-layer in the last block lifts the hidden dimension to the one of the context vector. A mean-pooling of the output matrix produces the context vector, which is then fed into the hypernetworks to generate model parameters.

C. HOTET for computing individual OT maps.

To demonstrate the work flow of HOTET, we first discuss the computation of individual maps between two distributions with HOTET (though HOTET is **suboptimal** in this case).

HOTET consists of three modules: the embedding network $\mathcal{E}$ and two hypernetworks $\mathcal{F}, \mathcal{G}$. Given a pair of empirical distributions μ, ν, the embedding network $\mathcal{E}$ is applied to generate context vectors $\mathcal{E}(\mu), \mathcal{E}(\nu)$ as specified in Section III-B. The two hypernetworks $\mathcal{F}, \mathcal{G}$ will then respectively take the context $\mathcal{E}(\mu), \mathcal{E}(\nu)$ as inputs, and produce parameters

$$\theta = \mathcal{F} \circ \mathcal{E}(\mu), \quad \omega = \mathcal{G} \circ \mathcal{E}(\nu),$$

for the two ICNNs $f_\theta(\cdot)$ and $g_\omega(\cdot)$ approximating the convex potentials. Afterwards, the potential networks are provided to the base OT solver, which concludes the entire forward pass. The resulting gradients of the forward pass are then used to update the three modules ($\mathcal{E}$ and $\mathcal{F}, \mathcal{G}$) with backpropagation.

As the parameter space for θ, ω in HOTET is strictly smaller than in the original solver, there will be natural concerns about the representation power of HOTET. Shortly in Section IV-B, we compare HOTET with the MMv2 solver as a sanity check, under the same setting taken by [29]; we note the proposed framework still achieves a comparable performance in unfavorable settings. The details are provided in Appendix A-C.

D. HOTET for computing multiple OT maps

We then illustrate the desired scenario for HOTET where the source measures $\mu_1, \ldots, \mu_n$ (training set) and the reference measure ν are already given and one aims to efficiently obtain the corresponding OT maps between a new μ and the reference ν. In training, instead of computing OT maps individually for the n distribution pairs, HOTET generates the $2n$ sets of parameters for ICNNs together via $\mathcal{F}$ and $\mathcal{G}$. To train the hypernetworks, loss from the base OT solver is computed for each (μ_i, ν) and subsequently aggregated together.

TABLE II: Cosine similarity between $\hat{T}$ and T^*. The standard deviation is calculated over 10 runs. Best viewed zoom in.

DIM	HOTET (fwd)	MetaOT (fwd)	MM-B (fwd)	HOTET (inv)	MetaOT (inv)	MM-B (inv)
2	0.93 ±0.01	0.81 ±0.01	0.99 ±0.01	0.97 ±0.01	0.74 ±0.01	0.99 ±0.01
4	0.87 ±0.01	0.84 ±0.01	0.96 ±0.01	0.96 ±0.01	0.88 ±0.01	0.97 ±0.01
8	0.89 ±0.01	0.80 ±0.01	0.94 ±0.01	0.97±0.01	0.82 ±0.01	0.97 ±0.01
16	0.90 ±0.01	0.74 ±0.01	0.92 ±0.01	0.97 ±0.01	0.78 ±0.01	0.96 ±0.01
32	0.91 ±0.01	0.76 ±0.01	0.90 ±0.01	0.91 ±0.01	0.80 ±0.01	0.97 ±0.01
64	0.84 ±0.01	0.71 ±0.01	0.90 ±0.01	0.91 ±0.01	0.87 ±0.01	0.94 ±0.01

TABLE III: Performance of the constructed transport forward (fwd) and inverse (inv) maps in predicting OT maps ($\mathcal{L}^2$-UVP (%) as the metric). The standard deviation is calculated over 10 runs.

DIM	Train				Predict			
	HOTET (fwd)	MetaOT (fwd)	HOTET (inv)	MetaOT (inv)	HOTET (fwd)	MetaOT (fwd)	HOTET (inv)	MetaOT (inv)
2	3.25 ±0.56	2.66 ±0.41	3.01 ±0.49	2.71 ±0.43	3.13 ±0.47	2.63 ±0.45	3.07 ±0.45	2.69 ±0.47
4	3.40 ±0.29	20.27 ±2.12	3.44 ±0.30	23.37 ±2.43	3.37 ±0.27	20.27 ±2.11	3.43 ±0.29	23.48 ±2.46
8	6.59 ±0.28	50.34 ±0.81	6.48 ±0.27	54.26 ±1.45	6.54 ±0.31	50.35 ±0.82	6.45 ±0.28	54.29 ±1.48
16	10.72 ±0.26	73.76 ±0.82	11.19 ±0.27	74.02 ±1.15	10.59 ±0.25	73.70 ±0.82	11.05 ±0.25	73.98 ±1.16
32	18.00 ±0.47	86.93 ±0.37	18.96 ±0.54	88.27 ±0.35	18.03 ±0.44	86.92 ±0.37	19.00 ±0.53	88.27 ±0.33
64	29.18 ±0.46	93.13 ±0.25	26.44 ±0.36	94.13 ±0.40	28.29 ±0.83	93.14 ±0.25	26.49 ±0.35	94.11 ±0.41

Since the reference measure ν is fixed in the training and testing process, we take μ_i as input to model **both the forward map and the inverse map** between μ_i and ν. That is, the parameters for the ICNNs f_{θ_i} (resp. g_{ω_i}) are generated as $\theta_i = \mathcal{F} \circ \mathcal{E}(\mu_i)$ (resp. $\omega_i = \mathcal{G} \circ \mathcal{E}(\mu_i)$)

IV. Numerical Experiments

We conducted various experiments to evaluate the performance of HOTET. The detailed experimental setups are discussed in Section IV-A. In particular, we perform a sanity check (Section IV-B) based on the benchmarks from [29] to exhibit the capability of our paradigm to generate quality transport map. This particular setting will be referred to as **W2B** hereinafter. Furthermore, we evaluated the prediction performance of our proposed training paradigm in Section IV-C, where we demonstrated that HOTET is capable of generating transport maps for unseen distributions after being trained on similar sample distributions (c.f. Section III-D). Lastly, we tested our model on images data through various applications in Section IV-D, and performed an ablation study on the embedding module in Section IV-E.

A. Experiment Settings

1) Choice of base OT solvers: While the theory foundation for OT is solid, the choice of a better solver for a specific task is an engineering problem. It is typical that in some cases the solvers will be hard to optimize: for example, the MMv2 solver tends to produce forward and inverse maps with large performance discrepancy, due to its asymmetric nature (detailed in Appendix A-B). When applicable, we compare the performance of the MM-B and MMv2 solver (see Appendix A-F), and report the results of the more stable one in the main text.

2) W2B benchmark: The performance of numerous OT solvers are evaluated in [29] on Gaussian mixture distributions. The performance of an estimated transport map $\hat{T}$ with respect to the ground truth T^* is evaluated with the $\mathcal{L}^2$-Unexplained Variance Percentage (UVP) and Cosine Similarity (CS):

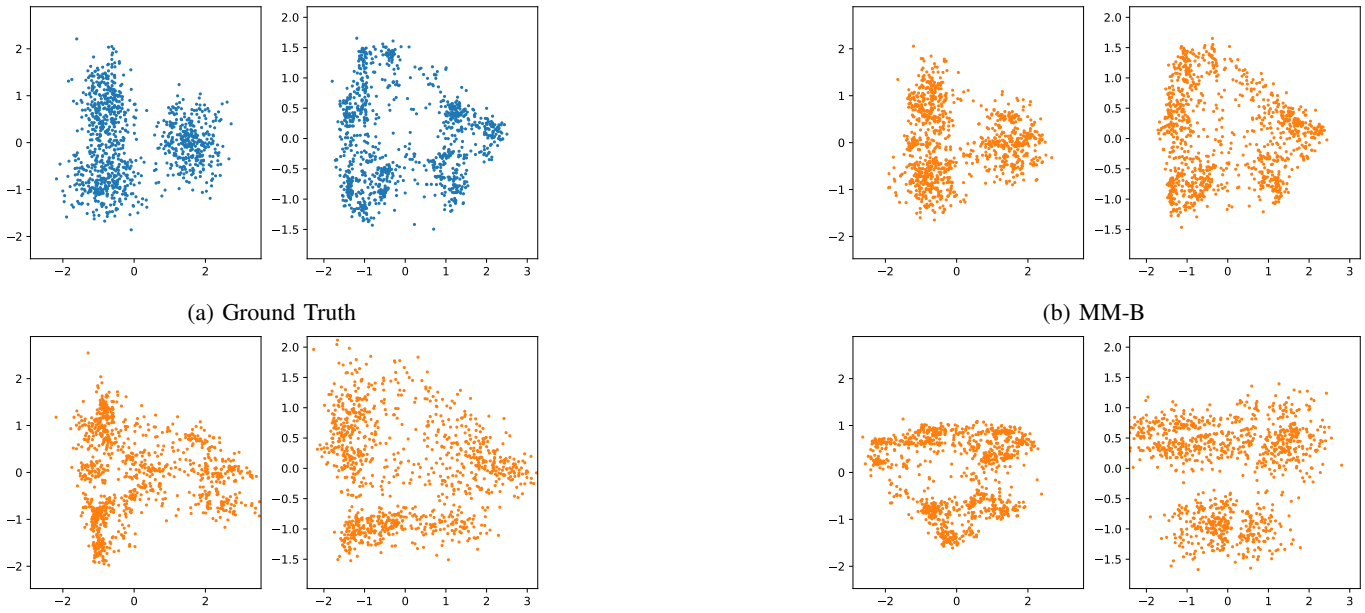
$$\mathcal{L}^2\text{-UVP}(\hat{T}) := 100 \cdot \frac{\|\hat{T} - T^*\|^2_{\mathcal{L}^2(\mu)}}{\text{Var}(\nu)}\%,$$

$$\text{CS}(\hat{T}) := \frac{\langle \hat{T} - \text{id}, \nabla\psi^* - \text{id} \rangle_{\mathcal{L}^2(\mu)}}{\|\hat{T} - \text{id}\|^2_{\mathcal{L}^2(\mu)} \cdot \|T^* - \text{id}\|^2_{\mathcal{L}^2(\mu)}}.$$

We inherited these settings for our own comparisons.

3) Predicting OT maps with HOTET: In this experiment, multiple Gaussian mixture distributions are generated with random means and covariance matrices. One distribution is designated as the reference, while the rest are divided into training and test sets. Distributions in the test set are used to assess the quality of the OT maps predicted by HOTET.

4) Color transfer: To further explore the capabilities of HOTET, we conducted color transfer experiments on paintings from WikiArt. The experiments include transferring colors from one image to another, as well as transferring colors from multiple images to a single target image.



(a) Ground Truth

(b) MM-B

(c) HOTET

(d) Meta OT

Fig. 2: Visualization of the transported data mapped via various methods. The ground truth data are shown in the top-left corner. For the other figures, the left (resp. right) subplots are generated by applying the forward maps $\hat{T}$ (resp. inverse maps $\hat{T}^{-1}$) to the corresponding ground truth data.

B. W2B Benchmark

We trained HOTET with the MM-B solver using the high-dimensional distribution dataset from W2B For the estimated forward and inverse transport maps, we examine the divergence between ν and $\hat{T}_{\#}\mu$, as well as the one between μ and, $\hat{T}_{\#}^{-1}\nu$, which are visualized in Figure 2. To make the comparison fair, the architectures of the ICNNs used in all the methods are identical.

The results are reported in Tables I and II, indicating that the transport maps generated by HOTET are comparable to those trained directly. This evidence validates that HOTET is able to generate quality transport maps.

C. Predict OT Maps with HOTET

We then evaluated the prediction performance of HOTET. In this experiment, Gaussian mixtures with three components, across multiple dimensions are generated, and each with different mean and covariance. One mixture ν was chosen as the reference, while 500 others formed the training set. After fitting 500 OT maps under HOTET, we then predicted the OT maps between ν and 100 new Gaussian mixtures (we found that MMv2 performed better in this setting than MM-B). For comparison, Meta OT was also trained in the setting above.

The results are summarized in Table III. The prediction performance of HOTET quickly exceeds baselines as the dimension increases, exhibiting the capability of HOTET to capture distribution embeddings. Due to space limit, evaluation on time efficiency is deferred to Appendix A-D.

Key note: removal of the pretraining stage. [29] suggested a pre-training stage to turn the ICNN into an identity map, before the regular training procedure. This process is time-consuming and unnecessary given the recent advance in transfer learning; we followed [39] and initialized the weights $W_h \sim \mathcal{N}(0, 0.1)$ in hypernetworks with a small variance. This way, we utilized the residual connection within ICNNs (with the tiny initial weights the ICNN already approaches an identity map) and thus can skip the pre-training stage.

D. Color Transfer

In this experiment, we used the OT map to transform the color histogram of an image to match that of another. The transport maps were constructed by interpreting the input RGB images as samples from their respective color distributions over the support $[0, 1]^3$. As before, we considered both one-to-one and many-to-one settings, employing the MM-B solver.

One-to-one color transfer. Given two color histograms, x and y, we trained HOTET to generate OT maps $\hat{T}, \hat{T}^{-1}$. To obtain new images, we replaced the colors of individual pixels so that the resulting color histograms became $x_{\text{trans}} = \hat{T}_{\#}x$ and $y_{\text{trans}} = \hat{T}_{\#}^{-1}y$, respectively. The results are shown in Figure 3.

Multiple-to-one color transfer. Given a collection of color histograms $X = \{x_1, x_2, \ldots, x_n\}$ and a reference histogram y, we trained HOTET to generate simultaneously the forward transport maps $\{\hat{T}_1, \hat{T}_2, \ldots, \hat{T}_n\}$ and inverse maps $\{\hat{T}_1^{-1}, \hat{T}_2^{-1}, \ldots, \hat{T}_n^{-1}\}$.

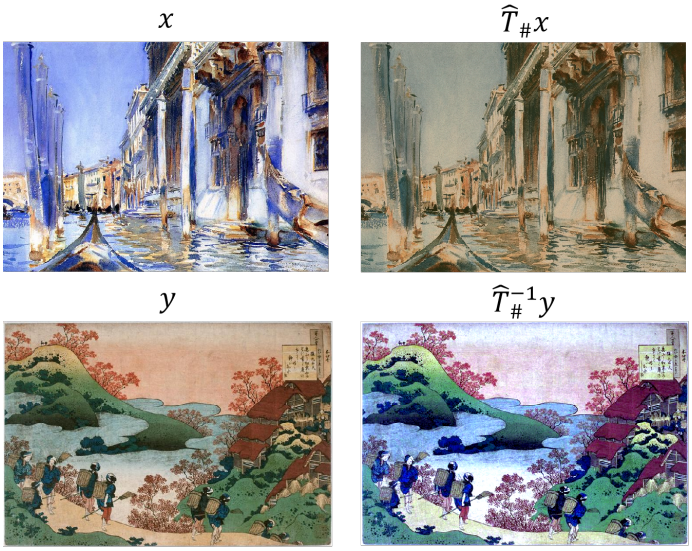


Fig. 3: One-to-one color transfer using HOTET.

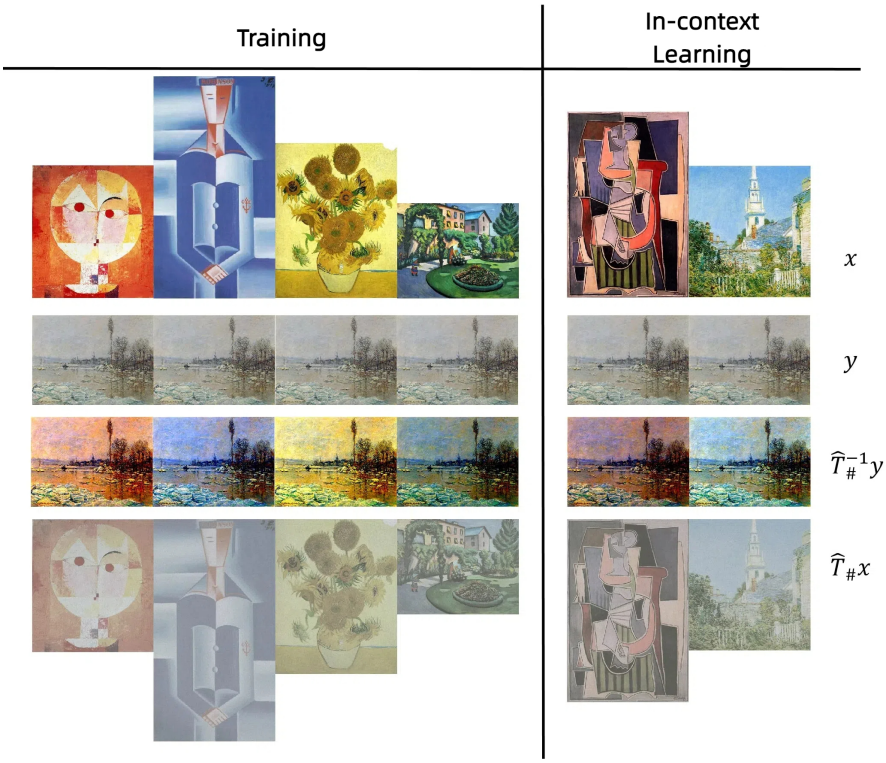


Fig. 4: Multiple-to-one color transfer using HOTET. Showing both the training and in-context learning stages.

The images with new color histograms, shown in the left panel of Figure 4, were generated in the same way as in the one-to-one setting.

In-context learning. To further evaluate the generalization capacity of a trained HOTET, we followed the in-context learning setting in [27] and reused the trained HOTET model from the multiple-to-one setting to generate OT maps for previously unseen color histograms x_{new}. Instead of a full training run (5000 iterations), we fine-tuned the HOTET on new images through only 50 warm-up steps. The results are presented in the right panel of Figure 4, demonstrating that the OT maps produced by short-term fine-tuning are visually comparable to those generated through complete training.

E. Ablation Studies on the Distribution Embedding Module

In this experiment, we **removed** the embedding module to investigate whether the overall performance would be impacted. We followed the setting in Section IV-C and the dimension is set at 64; the results are provided in Table IV. As expected,

TABLE IV: $\mathcal{L}^2$-UVP (%) results of the two different methods for obtaining the forward and inverse transport maps.

Method	Train		Predict	
	Fowrad Map	*Inverse Map*	*Fowrad Map*	*Inverse Map*
HOTET	29.18 ±0.46	26.44 ±0.36	28.29 ±0.83	26.49 ±0.35
No Emb.	82.75 ±0.55	69.42 ±0.47	83.71 ±0.51	69.54 ±0.64

HOTET significantly outperforms the version without embedding, highlighting the importance of the embedding module in effectively capturing signals from empirical distributions.

V. Conclusions and Limitations

In this paper, we recognize the increasing needs for computation of OT maps in modern signal processing, and propose a training paradigm HOTET to learn the OT maps between an unseen empirical distribution and a reference measure. In HOTET, information from the input distributions is extracted by a transformer, and then passed to a hypernetwork to generate the desired OT maps. Extensive experiments were conducted to demonstrate the efficacy of our new paradigm, showing that it is capable of producing quality OT maps.

Limitations: Despite the efficacy of HOTET, it faces challenges in the many-to-many setting, where data are multiple i.i.d. distribution pairs (μ_i, ν_i). Further exploration is needed to address this complex scenario.

VI. Acknowledgments

Mingchen Jiang was supported by the "R&D Hub Aimed at Ensuring Transparency and Reliability of Generative AI Models" project of the Ministry of Education, Culture, Sports, Science and Technology.

REFERENCES

[1] C. Villani, *Topics in Optimal Transportation*, ser. Graduate Studies in Mathematics. American Mathematical Society, 2003, vol. 58.
[2] V. M. Panaretos and Y. Zemel, *An invitation to statistics in Wasserstein space*. Springer Nature, 2020.
[3] Y. Polyanskiy and Y. Wu, "Wasserstein continuity of entropy and outer bounds for interference channels," *IEEE Transactions on Information Theory*, vol. 62, no. 7, pp. 3992–4002, 2016.
[4] Y. Cai and L.-H. Lim, "Distances between probability distributions of different dimensions," *IEEE Transactions on Information Theory*, vol. 68, no. 6, pp. 4020–4031, 2022.
[5] X. Cheng, J. Lu, Y. Tan, and Y. Xie, "Convergence of flow-based generative models via proximal gradient descent in wasserstein space," *IEEE Transactions on Information Theory*, vol. 70, no. 11, pp. 8087–8106, 2024.
[6] J. Fan and H.-G. Müller, "Conditional wasserstein barycenters and interpolation/extrapolation of distributions," *IEEE Transactions on Information Theory*, pp. 1–1, 2024.
[7] Y. Lipman, R. T. Q. Chen, H. Ben-Hamu, M. Nickel, and M. Le, "Flow matching for generative modeling," in *The Eleventh International Conference on Learning Representations*, 2023.
[8] X. Liu, C. Gong, and qiang liu, "Flow straight and fast: Learning to generate and transfer data with rectified flow," in *The Eleventh International Conference on Learning Representations*, 2023.
[9] T. Salimans, H. Zhang, A. Radford, and D. Metaxas, "Improving gans using optimal transport," *arXiv preprint arXiv:1803.05573*, 2018.
[10] T. Xu, L. K. Wenliang, M. Munn, and B. Acciaio, "Cot-gan: Generating sequential data via causal optimal transport," *Advances in neural information processing systems*, vol. 33, pp. 8798–8809, 2020.
[11] N. Kolkin, J. Salavon, and G. Shakhnarovich, "Style transfer by relaxed optimal transport and self-similarity," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2019, pp. 10 051–10 060.
[12] S. Kolouri, N. Naderializadeh, G. K. Rohde, and H. Hoffmann, "Wasserstein embedding for graph learning," in *International Conference on Learning Representations*, 2021.
[13] C. Moosmüller and A. Cloninger, "Linear optimal transport embedding: provable wasserstein classification for certain rigid transformations and perturbations," *Information and Inference: A Journal of the IMA*, vol. 12, no. 1, pp. 363–389, 09 2022.
[14] X. Lian, K. Jain, J. Truszkowski, P. Poupart, and Y. Yu, "Unsupervised multilingual alignment using wasserstein barycenter," in *Proceedings of the Twenty-Ninth International Conference on International Joint Conferences on Artificial Intelligence*, 2021, pp. 3702–3708.
[15] S. Alqahtani, G. Lalwani, Y. Zhang, S. Romeo, and S. Mansour, "Using optimal transport as alignment objective for fine-tuning multilingual contextualized embeddings," *arXiv preprint arXiv:2110.02887*, 2021.
[16] N. Courty, R. Flamary, and D. Tuia, "Domain adaptation with regularized optimal transport," in *Machine Learning and Knowledge Discovery in Databases: European Conference, ECML PKDD 2014*, 2014.
[17] N. Courty, R. Flamary, A. Habrard, and A. Rakotomamonjy, "Joint distribution optimal transportation for domain adaptation," *Advances in neural information processing systems*, vol. 30, 2017.
[18] S. P. Singh and M. Jaggi, "Model fusion via optimal transport," *Advances in Neural Information Processing Systems*, vol. 33, pp. 22 045–22 055, 2020.
[19] T. Wei, Z. Guo, Y. Chen, and J. He, "Ntk-approximating mlp fusion for efficient language model fine-tuning," in *International Conference on Machine Learning*. PMLR, 2023, pp. 36 821–36 838.
[20] M. Ai, T. Wei, Y. Chen, Z. Zeng, R. Zhao, G. Varatkar, B. D. Rouhani, X. Tang, H. Tong, and J. He, "Resmoe: Space-efficient compression of mixture of experts llms via residual restoration," in *31st SIGKDD Conference on Knowledge Discovery and Data Mining - Research Track*, 2025.
[21] D. Haviv, R. Z. Kunes, T. Dougherty, C. Burdziak, T. Nawy, A. Gilbert, and D. Pe'Er, "Wasserstein wormhole: Scalable optimal transport distance with transformers," *ArXiv*, pp. arXiv–2404, 2024.
[22] K. Kan, X. Li, and S. Osher, "Ot-transformer: A continuous-time transformer architecture with optimal transport regularization," *arXiv preprint arXiv:2501.18793*, 2025.
[23] M. Imfeld, J. Graldi, M. Giordano, T. Hofmann, S. Anagnostidis, and S. P. Singh, "Transformer fusion with optimal transport," 2024. [Online]. Available: https://arxiv.org/abs/2310.05719
[24] Z. L. Yaqing Chen and H.-G. Müller, "Wasserstein regression," *Journal of the American Statistical Association*, vol. 118, no. 542, pp. 869–882, 2023.
[25] N. Bonneel, G. Peyré, and M. Cuturi, "Wasserstein Barycentric Coordinates: Histogram Regression Using Optimal Transport," *ACM Transactions on Graphics*, vol. 35, no. 4, pp. 71:1–71:10, 2016.
[26] C. Bunne, A. Krause, and M. Cuturi, "Supervised training of conditional monge maps," *Advances in Neural Information Processing Systems*, vol. 35, pp. 6859–6872, 2022.
[27] B. Amos, S. Cohen, G. Luise, and I. Redko, "Meta optimal transport," *arXiv preprint arXiv:2206.05262*, 2022.
[28] A. Gracyk and X. Chen, "GeONet: a neural operator for learning the Wasserstein geodesic," *The Conference on Uncertainty in Artificial Intelligence (UAI)*, 2024.
[29] A. Korotin, L. Li, A. Genevay, J. M. Solomon, A. Filippov, and E. Burnaev, "Do neural optimal transport solvers work? a continuous wasserstein-2 benchmark," *Advances in Neural Information Processing Systems*, vol. 34, 2021.
[30] F. Santambrogio, *Optimal Transport for Applied Mathematicians*. Birkhäuser Cham, 2015.
[31] B. Amos, L. Xu, and J. Z. Kolter, "Input convex neural networks," in *International Conference on Machine Learning*. PMLR, 2017, pp. 146–155.
[32] D. Ha, A. M. Dai, and Q. V. Le, "Hypernetworks," in *International Conference on Learning Representations*, 2017.
[33] V. K. Chauhan, J. Zhou, P. Lu, S. Molaei, and D. A. Clifton, "A brief review of hypernetworks in deep learning," 2023.
[34] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, L. Kaiser, and I. Polosukhin, "Attention is all you need," in *Advances in Neural Information Processing Systems*, 2017.
[35] C. Yun, S. Bhojanapalli, A. S. Rawat, S. Reddi, and S. Kumar, "Are transformers universal approximators of sequence-to-sequence functions?" in *International Conference on Learning Representations*, 2020.
[36] T. Dao, D. Y. Fu, S. Ermon, A. Rudra, and C. Ré, "FlashAttention: Fast and memory-efficient exact attention with IO-awareness," in *Advances in Neural Information Processing Systems*, 2022.
[37] T. Dao, "FlashAttention-2: Faster attention with better parallelism and work partitioning," 2023, technical Report.
[38] J. Lee, Y. Lee, J. Kim, A. Kosiorek, S. Choi, and Y. W. Teh, "Set transformer: A framework for attention-based permutation-invariant neural networks," in *International conference on machine learning*. PMLR, 2019, pp. 3744–3753.
[39] N. Houlsby, A. Giurgiu, S. Jastrzebski, B. Morrone, Q. De Laroussilhe, A. Gesmundo, M. Attariyan, and S. Gelly, "Parameter-efficient transfer learning for nlp," in *International conference on machine learning*. PMLR, 2019, pp. 2790–2799.
[40] A. Mallasto, J. Frellsen, W. Boomsma, and A. Feragen, "(q, p)-wasserstein gans: Comparing ground metrics for wasserstein gans," *arXiv preprint arXiv:1902.03642*, 2019.
[41] A. Makkuva, A. Taghvaei, S. Oh, and J. Lee, "Optimal transport mapping via input convex neural networks," in *Proceedings of the 37th International Conference on Machine Learning*. PMLR, 2020.
[42] A. Korotin, V. Egiazarian, A. Asadulaev, A. Safin, and E. Burnaev, "Wasserstein-2 generative networks," in *International Conference on Learning Representations*, 2021.
[43] R. T. Rockafellar, *Convex Analysis*. Princeton University Press, 1970.

[44] Y. LeCun, L. Bottou, Y. Bengio, and P. Haffner, "Gradient-based learning applied to document recognition," *Proceedings of the IEEE*, vol. 86, no. 11, pp. 2278–2324, 1998.
[45] D. P. Kingma and M. Welling, "Auto-encoding variational bayes," *arXiv preprint arXiv:1312.6114*, 2013.
[46] X. L. Li and P. Liang, "Prefix-tuning: Optimizing continuous prompts for generation," in *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*. Online: Association for Computational Linguistics, Aug. 2021, pp. 4582–4597.
[47] K. M. Choromanski, V. Likhosherstov, D. Dohan, X. Song, A. Gane, T. Sarlos, P. Hawkins, J. Q. Davis, A. Mohiuddin, L. Kaiser, D. B. Belanger, L. J. Colwell, and A. Weller, "Rethinking attention with performers," in *International Conference on Machine Learning*. PMLR, 2021.
[48] Y. Chen, Q. Zeng, H. Ji, and Y. Yang, "Skyformer: Remodel self-attention with gaussian kernel and nyström method," *Advances in Neural Information Processing Systems*, vol. 34, pp. 2122–2135, 2021.
[49] L. Wasserman, *All of Nonparametric Statistics*. Springer New York, 2006.

Appendix to "Embedding Empirical Distributions for Computing Optimal Transport Maps"

APPENDIX A
MORE ON EXPERIMENTS

A. Implementation details and discussions

Despite the theoretical results for OT, there are a few numerical issues in implementing the HOTET framework, which significantly impact the empirical performance without proper practical adaptation.

1) Asymmetry in computing multiple OT maps w.r.t. one reference measure: In the setting of Section III-D, the roles of the two hypernetworks differ: the hypernetwork $\mathcal{G}$ for inverse maps still depends on μ_i's, and is expected to precisely transport the single reference measure ν onto different destinations. Due to the inherent asymmetry in the MMv2 solver, practically the forward maps φ_i usually show superior performance over the inverse ones.

We take the special characteristics into consideration when devising the implementation of HOTET. Specifically, for the OT maps required in downstream applications, such as Wasserstein embedding or color transfer, we suggest setting them as the forward maps, to obtain high-quality neural maps.

2) Restriction on parameters of ICNN: Recall from Section II-B that the parameters of certain weight matrices in Equation (B.1) must remain non-negative. To enforce this restriction, we adopt *projected gradient descent* and formally apply and additional ReLU activation to the selected outputs of the hypernetwork; other alternatives, such as Softplus, are evaluated while we found ReLU is numerically the most stable choice.

B. Implementation of the Base OT Solvers

In this work, we considered two base OT solvers: MM-B [40] MMv2 [41]. Both solvers are proven to perform reasonably well via benchmarking [29].

a) MM-B Solver: The MM-B solver is built upon a reformulation [1, Thm. 2.9] of the dual problem (DP). By dropping the constant terms, the problem reduces to an equivalent formulation [42, Eqn. 5]:

$$\min_{\varphi\in\mathcal{C}}\left\{\int\varphi(x)\,d\mu(x)+\int\varphi^{\dagger}(y)\,\mathrm{d}\nu(y)\right\},\qquad \varphi^{\dagger}(y)\coloneqq\sup_{x\in\mathbb{R}^d}\left\{\langle x,y\rangle-\varphi(x)\right\}. \tag{Cor}$$

where $\varphi^{\dagger}$ is the Fenchel conjugate [43] of φ, and $\mathcal{C}$ denotes the class of convex functions. The main challenge in computing $\varphi^{\dagger}$ lies in finding $x(y)$ that achieves the maximum for each y on the support of ν. The MM-B solver simplifies this process by addressing the inner problem only on minibatches sampled from μ and ν. Specifically, let f_θ be an ICNN, the MM-B solver, as implemented by [29], optimize θ as follows: given minibatches $X\sim\mu$ and $Y\sim\nu$ of size B, the loss function to be minimized is computed as

$$L(\theta)=\frac{1}{B}\sum_{j=1}^{B}f_\theta(X_j)-f_\theta\big(X_{i(j)}\big),\qquad i(j)\coloneqq\arg\max_{i\in[B]}\langle X_i,Y_j\rangle-f_\theta(X_i).$$

Although this approach produces a biased solution, it significantly accelerates computation.

To solve for φ and its conjugate simultaneously, the loss function can be symmetrized as

$$L(\theta,\omega)=L(\theta)+\left(\frac{1}{B}\sum_{j=1}^{B}g_\omega(Y_j)-g_\omega\big(Y_{k(j)}\big)\right),\qquad k(j)\coloneqq\arg\max_{k\in[B]}\langle Y_k,X_j\rangle-g_\omega(Y_k).$$

where g_ω is an ICNN that approximates the conjugate of f_θ.

b) MMv2 Solver: The MMv2 solver, on the other hand, is based on the reformulation [41, Thm. 3.3]

$$\min_{\varphi\in\mathcal{C}} \int \varphi(x)\, d\mu(x) + \int \max_{\psi\in\mathcal{C}} \{\langle \nabla\psi(y), y\rangle - \varphi\circ\nabla\psi(y)\}\, \mathrm{d}\nu(y).$$

To compute the OT map, potentials φ and ψ are approximated by ICNNs f_θ and g_ω, respectively. The parameters θ and ω are updated in an alternating fashion. For the inner problem, ω is updated through maximizing

$$L(\omega) = \frac{1}{B}\sum_{i=1}^{B} \langle \nabla g_\omega(Y_i), Y_i\rangle - f_\theta\circ\nabla g_\omega(Y_i)$$

via SGD, with multiple iterations performed to ensure convergence. Afterward, θ is updated by minimizing

$$L(\theta) = \frac{1}{B}\sum_{i=1}^{B} f_\theta(X_i) - f_\theta\circ\nabla g_\omega(Y_i).$$

This procedure is repeated until both θ and ω converge to near-optimal values.

C. Pseudocode for Network Training

Algorithm 1 outlines the training process for HOTET using the MM-B solver. Here, $\mathcal{X} \coloneqq \{X_i\}_{i=0}^n$ represents the set of distributions, and Y is the reference distribution. In the case where $\mathcal{X}$ contains only a single distribution, it reduces to a classical one-to-one OT problem. For scenarios where $n > 1$ (e.g. OT maps prediction), a batch of distributions, with size $B \leq n$, is sampled from $\mathcal{X}$. Then, from each sampled distribution and the reference y, batches of size b are drawn for evaluating the individual OT losses. The networks f_θ and g_ω are ICNNs that parameterize the convex dual potentials. The hypernetworks $\mathcal{F}$ and $\mathcal{G}$ generate the parameters for these potential networks, while $\mathcal{E}$ serving as the embedding module. The variable K denotes the total number of training iterations.

TABLE V: Time cost (sec) of training different models. The batch sizes in the 32/64 dimensional settings are 256, and in other settings are 1024. The GPU is a single Nvidia RTX 4090.

Model	2	4	8	16	32	64
MetaOT	1161	1183	1214	1248	1405	4198
HOTET-MMB	1228	1239	1339	1405	1223	2324
HOTET-MMv2	30972	31762	29106	30541	28150	20120
MM-B	20695	21848	24099	24615	19618	52633
MMv2	409284	365529	418475	411523	401511	395583

D. Runtime analysis

We further assessed the time efficiency of HOTET, MetaOT, and repeating MMv2 solver in an 8-dimensional setting. The results in Table V showed that HOTET performs similarly to MetaOT. However, training networks with MMv2 solver directly is more time-consuming, as it requires training each distribution pair individually 500 times in this setting. Meanwhile, a direct MMv2 solver does not have prediction capabilities, as the potentials of the trained networks only represents the transport maps between the two input distributions.

E. Additional Experiment Results

Figures 5 and 6 present OT maps learned in the W2B experiment (Section IV-B) with $d = 4$ and $d = 8$. The results are projected onto the first 2 principal directions for visualization.

Algorithm 1 Training Procedure of HOTET with MM-B Solver

procedure TRAINMODEL($\mathcal{X}, Y, f_\theta, g_\omega, \mathcal{F}, \mathcal{G}, \mathcal{E}$)
  Initialize the parameters of each module
  for $\mathtt{t} = 1, \ldots, K$ **do**
    Sample batch $\mathcal{X}_{\text{batch}}$ of size $\mathtt{B}$ from $\mathcal{X}$.
    $\mathcal{L} \leftarrow 0$
    for $\mathtt{i} = 1, \ldots, \mathtt{B}$ **do**
      Sample batches $\mathtt{x}$, $\mathtt{y}$ of size $\mathtt{b}$ from X_i, Y, respectively
      $\mathtt{embedding\_x} \leftarrow \mathcal{E}(\mathtt{x})$
      $\mathtt{embedding\_y} \leftarrow \mathcal{E}(\mathtt{y})$
      $\mathcal{L}_{xy} \leftarrow$ COMPUTELOSSFORWARD($\mathtt{x}, \mathtt{y}, \mathcal{F}, \mathtt{embedding\_x}$)
      $\mathcal{L}_{yx} \leftarrow$ COMPUTELOSSINVERSE($\mathtt{y}, \mathtt{x}, \mathcal{G}, \mathtt{embedding\_y}$)
      $\mathcal{L} \leftarrow \mathcal{L} + (\mathcal{L}_{xy} + \mathcal{L}_{yx})/(2 \cdot \mathtt{B})$
    end for
    Update model $\mathcal{E}, \mathcal{F}, \mathcal{G}$ according to $\mathcal{L}$
  end for
end procedure

function COMPUTELOSSFORWARD($\mathtt{x}, \mathtt{y}, \mathcal{F}, \mathtt{embedding\_x}$)
  $\theta \leftarrow \mathcal{F}(\mathtt{embedding\_x})$
  $\mathtt{x\_push} \leftarrow \nabla f(\mathtt{x} \mid \theta)$
  $\mathtt{xy} \leftarrow \langle \mathtt{x}, \mathtt{y} \rangle$
  $\mathtt{idx\_y} \leftarrow \text{argmax}(\mathtt{xy} - \mathtt{x\_push}, \text{dim} = 0)$
  $\mathtt{y\_push} \leftarrow \mathtt{x[idx\_y]}$
  $\mathtt{W\_loss\_xy} \leftarrow \text{mean}(\mathtt{x\_push} - \nabla f(\mathtt{y\_push} \mid \theta))$
end function

function COMPUTELOSSINVERSE($\mathtt{y}, \mathtt{x}, \mathcal{G}, \mathtt{embedding\_y}$)
  $\omega \leftarrow \mathcal{G}(\mathtt{embedding\_y})$
  $\mathtt{y\_push} \leftarrow \nabla g(\mathtt{y} \mid \omega)$
  $\mathtt{yx} \leftarrow \langle \mathtt{y}, \mathtt{x} \rangle$
  $\mathtt{idx\_x} \leftarrow \text{argmax}(\mathtt{yx} - \mathtt{y\_push}, \text{dim} = 0)$
  $\mathtt{x\_push} \leftarrow \mathtt{y[idx\_x]}$
  $\mathtt{W\_loss\_yx} \leftarrow \text{mean}(\mathtt{y\_push} - \nabla g(\mathtt{x\_push} \mid \omega))$
end function

TABLE VI: Performance of the constructed forward (fwd) and inverse (inv) transport maps by MM-B and MMv2 solvers in W2B benchmark ($\mathcal{L}^2$-UVP (%) as the metric). Lower implies the fitted map approximates the true OT map better.

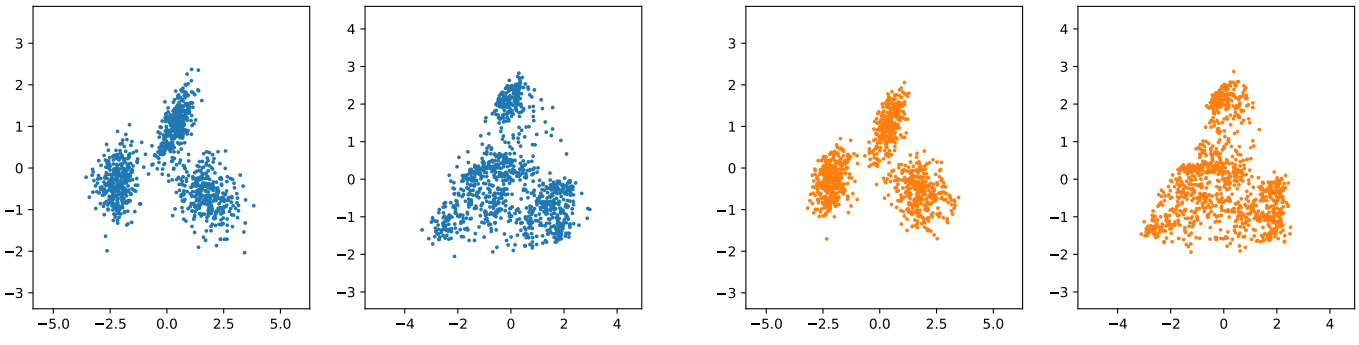
DIM	HOTET-MMB (fwd)	HOTET-MMv2 (fwd)	HOTET-MMB (inv)	HOTET-MMv2 (inv)
2	5.03 ± 0.33	13.34 ± 0.46	10.49 ± 0.52	5.31 ± 0.58
4	10.06 ± 0.39	30.74 ± 0.55	15.54 ± 0.46	23.84 ± 0.67
8	11.38 ± 0.41	33.90 ± 0.61	18.79 ± 0.45	16.92 ± 0.52
16	16.91 ± 0.58	30.86 ± 0.56	25.62 ± 0.49	28.72 ± 0.39
32	20.87 ± 0.56	40.72 ± 0.54	23.04 ± 0.53	35.71 ± 0.47
64	37.62 ± 0.84	38.42 ± 0.51	22.94 ± 0.87	36.06 ± 0.62

F. Choosing the Solvers

MM-B and MMv2 perform differently in the experiment settings in Section IV-B and Section IV-C, the details are in Table VI and Table VII. Therefore, we choose the better performed one in our main paper.

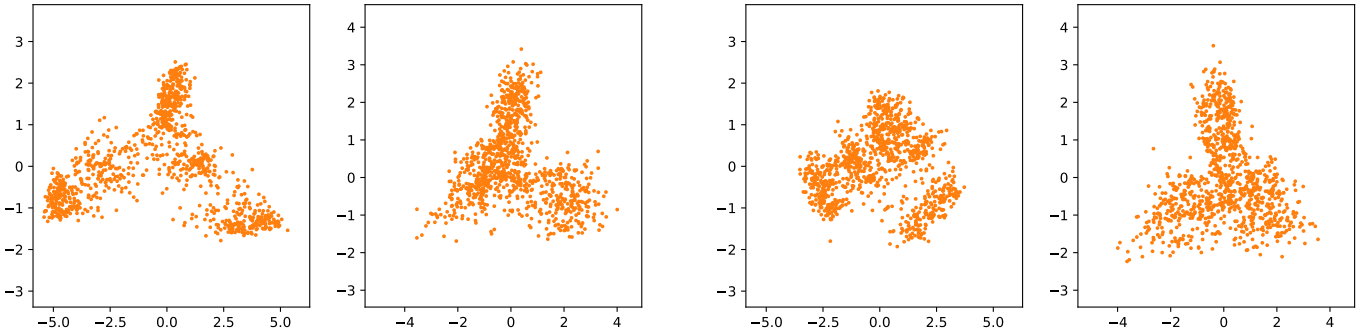
G. Validating the Embedding Module with the MNIST Dataset

In this experiment, we examined the embeddings produced by the embedding module $\mathcal{E}$ from a HOTET trained on the MNIST [44] handwritten digits dataset. We selected 6000 images from the training set (600 per digit), and then trained a



(a) Ground Truth

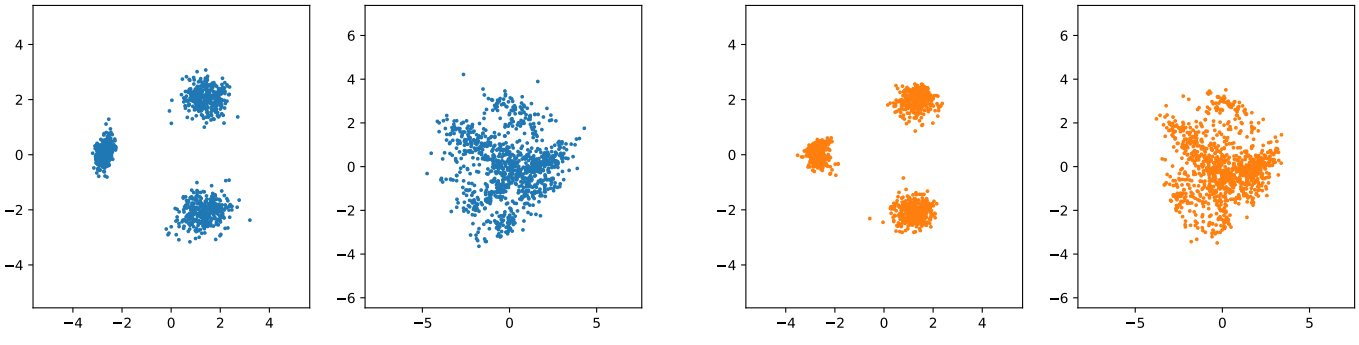
(b) MM-B



(c) HOTET

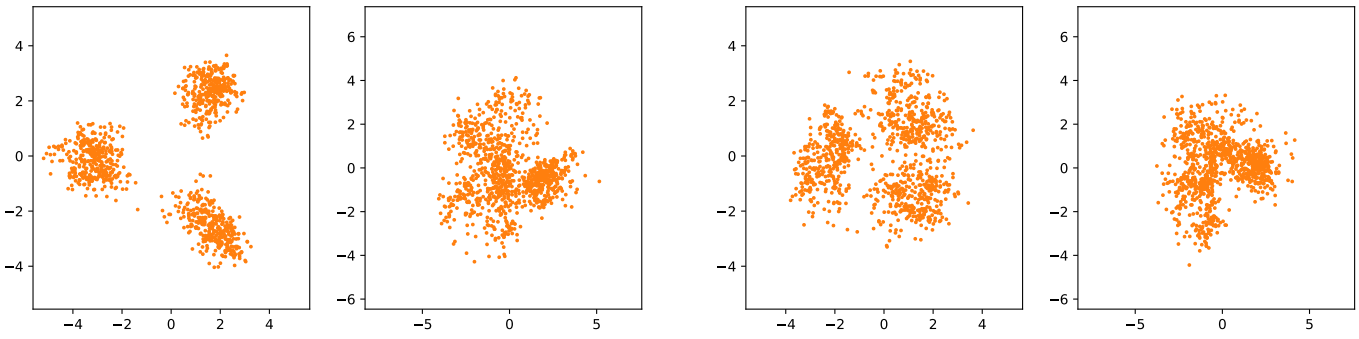
(d) Meta OT

Fig. 5: Samples generated by the forward and inverse maps in $d = 4$, compared with ground truth input distributions.



(a) Ground Truth

(b) MM-B



(c) HOTET

(d) Meta OT

Fig. 6: Samples generated by the forward and inverse maps in $d = 8$, compared with ground truth input distributions.

VAE [45] to map them to 6000 3-dimensional latent distributions. Next, we trained a HOTET with embedding dimension $d = 128$ to learn OT maps between the latent distributions and the reference measures $N(\mathbf{0}, \boldsymbol{I}_3)$. Afterwards, we visualized

TABLE VII: Performance of the constructed forward (fwd) and inverse (inv) transport maps by MM-B and MMv2 solvers in predicting OT maps setting ($\mathcal{L}^2$-UVP (%) as the metric). Lower implies the fitted map approximates the true OT map better.

DIM	Train				Predict			
	HOTET-MMv2 (fwd)	HOTET-MMB (fwd)	HOTET-MMv2 (inv)	HOTET-MMB (inv)	HOTET-MMv2 (fwd)	HOTET-MMB (fwd)	HOTET-MMv2 (inv)	HOTET-MMB (inv)
2	3.25 ± 0.56	4.23 ± 0.54	3.01 ± 0.49	4.03 ± 0.52	3.13 ± 0.47	4.12 ± 0.62	3.07 ± 0.45	3.84 ± 0.61
4	3.40 ± 0.29	4.95 ± 0.61	3.44 ± 0.30	3.73 ± 0.29	3.37 ± 0.27	3.57 ± 0.33	3.43 ± 0.29	3.66 ± 0.36
8	6.59 ± 0.28	6.95 ± 0.28	6.48 ± 0.27	7.00 ± 0.28	6.54 ± 0.31	6.80 ± 0.20	6.45 ± 0.28	6.94 ± 0.22
16	10.72 ± 0.26	12.60 ± 0.26	11.19 ± 0.27	12.64 ± 0.31	10.59 ± 0.25	12.66 ± 0.28	11.05 ± 0.25	12.71 ± 0.31
32	18.00 ± 0.47	20.36 ± 0.26	18.96 ± 0.54	20.35 ± 0.25	18.03 ± 0.44	20.00 ± 0.22	19.00 ± 0.53	19.93 ± 0.21
64	29.18 ± 0.46	28.69 ± 0.18	26.44 ± 0.36	28.72 ± 0.18	28.29 ± 0.83	28.74 ± 0.18	26.49 ± 0.35	28.84 ± 0.17

the training set embeddings using t-SNE to assess whether the embedding module effectively captured the information from the original images. The result is showed in Figure 7.

H. Figure Result of Individual OT Maps

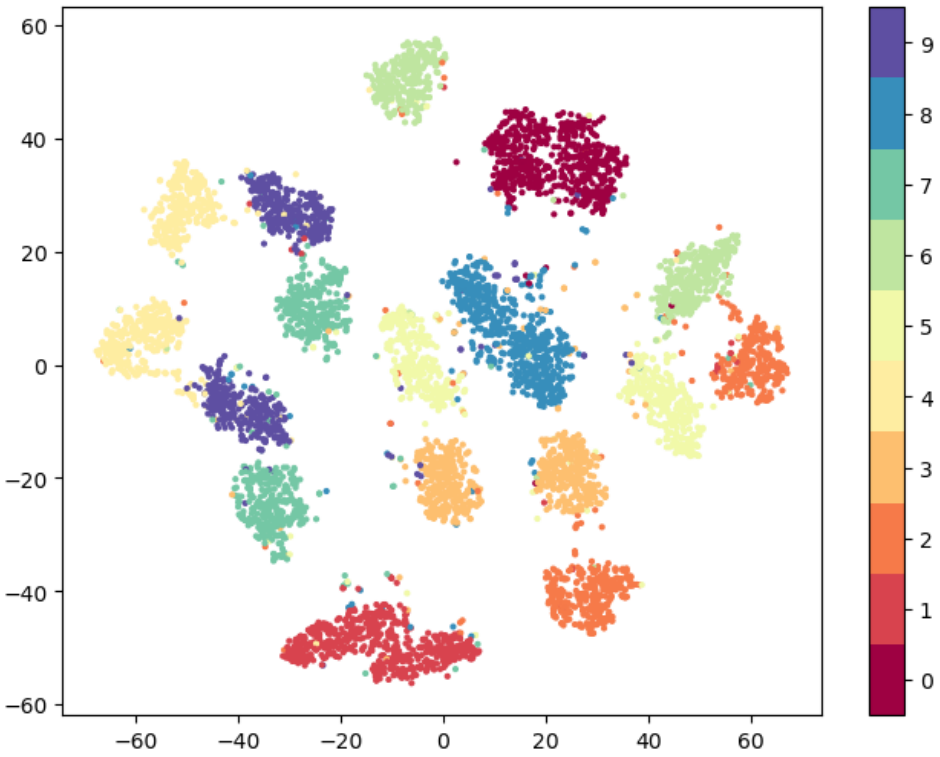


Fig. 7: A t-SNE visualization of the 128-dimensional vectors produced by the embedding module for MNIST, with the true labels used to color the points. The embeddings of the same digit clearly clustered together, indicating that the embedding module effectively preserved the information from the original images.

APPENDIX B
USEFUL FACTS

A. ICNN

The fully connected ICNN is a feed-forward neural network whose intermittent layer z_ℓ is the activation of the linear transformation of the previous layer, plus the affine transformation of the input $z_0 := x$. In other words, for $\ell \in \{1, \dots, L\}$,

$$z_\ell := \sigma_\ell(A_{\ell-1} z_{\ell-1} + W_{\ell-1} x + b_{\ell-1}), \tag{B.1}$$

where A is a non-negative matrix, W, b are regular unrestricted weight matrix and bias vector, and σ is a convex and non-decreasing activation function. The final ICNN output is $\tilde{\varphi}(x) = z_L$ for some pre-specified $L \geq 1$.

The structure of ICNN is justified by the following facts:

- The composition of a convex and non-decreasing function and a convex function is convex.
- The composition of a convex function and an affine function is convex.
- The non-negative sum of convex functions is also convex.

B. Transformer

Transformers [34] are a type of neural network architecture primarily used in natural language processing (NLP) tasks. They are composed by L stacked layers, where each layer comprises of a multi-headed attention and a fully connected feed-forward network (FFN) sub-layer. The attention sub-layer, assuming h heads and dimension size d for each head, first maps an input $X \in \mathbb{R}^{n \times hd}$ into the query (Q), key (K), and value (V) matrices through the following affine transformations:

$$Q/K/V = XW_{[q/k/v]} + \mathbf{1}b_{[q/k/v]}^{\top}, \tag{B.2}$$

where $Q, K, V \in \mathbb{R}^{n \times hp}$, W_q, W_k, W_v are $hd \times hd$ weight matrices, and $b_q, b_k, b_v \in \mathbb{R}^{N_h p}$ are the bias terms [1]. After the transformation, the three components Q, K, V are split into h blocks corresponding to different heads. For example, Q is re-written as $Q = (Q^{(1)}, \ldots, Q^{(N_h)})$, where each block $Q^{(h)} = XW_q^{(h)} + 1(b_q^{(h)})^T$ is an $n \times p$ matrix, and $W_q^{(h)}, b_q^{(h)}$ are the corresponding parts in W_q, b_q. The attention output for the h^{th} head is then computed as:

$$\boldsymbol{L}^{(h)}\boldsymbol{V}^{(h)} := \mathrm{softmax}(\boldsymbol{Q}^{(h)}(\boldsymbol{K}^{(h)})^T/\sqrt{p})\boldsymbol{V}^{(h)} = (\boldsymbol{D}^{(h)})^{-1}\boldsymbol{M}^{(h)}\boldsymbol{V}^{(h)}, \tag{B.3}$$

where $M^{(h)} := \exp(Q^{(h)}(K^{(h)})^T/\sqrt{p})$ and $D^{(h)}$ is a diagonal matrix in which $D_{ii}^{(h)}$ is the sum of the i-th row in $M^{(h)}$, corresponding to the normalization part in softmax.

After we obtain the outputs in each head, they are concatenated as,

$$L := \left(L^{(1)}V^{(1)}, \ldots, L^{(N_h)}V^{(N_h)}\right), \tag{B.4}$$

followed by the overall output,

$$LW_o + 1b_o^T, \tag{B.5}$$

where W_o and b_o are similarly sized as the other matrices in Equation (B.2).

C. Attention as Kernel Estimators

For each head in the attention module, we have given the expression of attention output in Equation B.3. In this subsection, we will re-write attention as a kernel estimator to show the connection.

In computing the attention output (of a single head), we have an input sequence $\{x_i\}_{i=1}^n$ (the rows in X) and accordingly we can obtain N [2] key vectors $\{k_j\}_{j=1}^N \subset \mathbb{R}^p$ (from the *key* matrix $\boldsymbol{K}$) and query vectors $\{q_i\}_{i=1}^n \subset \mathbb{R}^p$ (from $\boldsymbol{Q}$).[3] The original goal of self-attention is to obtain the representation of each input token x_i: $g(x_i)$. By denotation exchange: $q_i := x_i$ and $f(q_i) := g(x_i)$, we can also understand the aforementioned self-attention module as returning the representation $f(q_i)$ of

[1] To ease the notations we adopt the setting where X, Q, K, V have the same shape.
[2] Note that N may not always equal n, such as in cross attention ($N \neq n$) or in prefix-tuning ($N > n$ due to the prefix pretended to the key matrix) [46].
[3] In this subsection we omit the superscript (h) for simplicity since the discussion is limited within a single head

the input query vector q_i through $\{k_j\}_{j=1}^n$, which behaves as a kernel estimator [47], [48]. Specifically, for a single query vector q_i, a Nadaraya–Watson kernel estimator [49, Definition 5.39] models its representation as,

$$f(q_i) = \sum_{j=1}^{n} \ell_j(q_i) c_j, \quad \text{where} \quad \ell_j(q_i) \coloneqq \frac{\kappa(q_i, k_j)}{\sum_{j'=1}^{N} \kappa(q_i, k_{j'})}. \tag{B.6}$$

Here, $\kappa(\cdot,\cdot)$ is a kernel function, and c_j's are the coefficients (c_j can either be a scalar or a vector in different applications) that are learned during training. In this estimator, $\{k_j\}_{j=1}^n$ serve as the *supporting points* which help construct the representation for an input q_i.

For kernel function $\kappa(x,y) = \exp\left(\langle x, y\rangle / \sqrt{p}\right)$, we slightly abuse the notation $\kappa(\boldsymbol{Q}, \boldsymbol{K})$ to represent an n-by-N empirical kernel matrix, whose element in the i-th row and the j-th column is $\kappa(q_i, k_j), \forall i \in [n], j \in [N]$. With these notations, the representation of the transformed output will be,

$$\boldsymbol{D}^{-1} \kappa(\boldsymbol{Q}, \boldsymbol{K}) \boldsymbol{C}, \tag{B.7}$$

where $\boldsymbol{D}$ is a diagonal matrix for row normalization in Eq. (B.6), and $\boldsymbol{C}$ is an N-by-p matrix whose j-th row is c_j.

D. Potential Pathways to the Incorporation of Sample Weights into Distribution Embedding

Considering the correspondence between Equation (B.7) and the standard softmax attention in Equation (B.3), we are indeed able incorporate the sample weights in an empirical distribution to attention (as mentioned in Section III-B). The new character will allow a transformer to embed arbitrary empirical distributions and generate layer embeddings for hypernetworks.

Originally in transformers, all the tokens are assumed to share the equal weights, while a general empirical distribution allows non-uniform atom masses. To address the issue, we can make an analogy between self-attention and the Nadaraya–Watson kernel estimator in Equation (B.6).

We first rewrite $\ell_j(q_i)$'s in Equation (B.6) to highlight the implicitly assumed uniform sample weights:

$$\ell_j(q_i) = \frac{\kappa(q_i, k_j)}{\sum_{j'=1}^{N} \kappa(q_i, k_{j'})} = \frac{\frac{1}{N} \cdot \kappa(q_i, k_j)}{\sum_{j'=1}^{N} \frac{1}{N} \cdot \kappa(q_i, k_{j'})},$$

which allows an immediate extension in the weighted case; given the normalized sample weights $\boldsymbol{m} = \{m_1, m_2, \ldots, m_n\}$ with $\sum_{j=1}^{N} m_j = 1$, we can modify the coefficients $\ell_j(q_i)$'s in Equation (B.6) as

$$\ell_j(q_i) = \frac{m_j \cdot \kappa(q_i, k_j)}{\sum_{j'=1}^{N} m_{j'} \cdot \kappa(q_i, k_{j'})}.$$

Ultimately, the weighted attention is expressed as follows:

$$\boldsymbol{D}^{-1} \exp\left(\frac{\boldsymbol{Q}\boldsymbol{K}^T}{\sqrt{p}}\right) \operatorname{diag}\left(N\boldsymbol{m}\right) \boldsymbol{V} = \operatorname{softmax}\left(\frac{\boldsymbol{Q}\boldsymbol{K}^T}{\sqrt{p}} \operatorname{diag}\left(\ln N\boldsymbol{m}\right)\right) \boldsymbol{V},$$

where the row normalization matrix $\boldsymbol{D}$ is reloaded as $\operatorname{diag}\left(\exp\left(\frac{\boldsymbol{Q}\boldsymbol{K}^T}{\sqrt{p}}\right) N\boldsymbol{m}\right)$.

For the output $\boldsymbol{H}$ of the whole transformer, we can apply a weighted average pooling to obtain the final embedding:

$$\boldsymbol{z} = \boldsymbol{H}^T \boldsymbol{m}. \tag{B.8}$$

The embedding $\boldsymbol{z}$ will then be passed to the hypernetwork for generating the transport map.

E. Architecture Comparison with Existing Methods

We noted that our proposed paradigm shared some similarities with the general ideas of CondOT and Meta OT. Therefore, we shall briefly discuss the differences between our methods and theirs.

The setting of CondOT is based on a regression formulation. Therefore, the input source and target measures must be paired. Our scheme is more flexible compared to theirs in this regard, as it allows the numbers of source and target measures to differ. Further, the transport maps in CondOT are modulated by having an additional context variable as an input, while our transport maps are generated from properly trained hypernetworks. Lastly, we devised an end-to-end pipeline to extract information from source measures by transformers, while CondOT relied on externally given information to obtain the context variables.

The Meta OT model, on the other hand, makes use of hypernetworks to generate the transport maps. Therefore, it is more comparable to our proposed method. In Meta OT, however, the distributions are passed directly to the hypernetworks as inputs, which means they must be concatenated, padded, or resized if their size mismatch. The transformer module in HOTET resolves this matter and extract the information more efficiently, as explained in Section III-B.

Appendix C
Miscellanies

A. Notations

We denote by $\mathcal{E}$ the embedding module, and $\mathcal{F}, \mathcal{G}$ the hynernetworks in HOTET, respectively. Given distributions μ and ν, we use $T_{\mu\to\nu}$ to denote the true OT map that pushforwards μ to ν, and omit the subscript when the context is clear. Accordingly, we use φ, ψ to denote the potential functions and f, g to denote the networks used for approximating φ, ψ.

B. Hyperparameters

For the detail settings, Tables VIII to X show the hyperparameters we used in our experiments.

Model	LR	Batch Size	Total Iterations
HOTET	10^{-3}	1024	5000
MetaOT	10^{-3}	1024	5000
MM-B	10^{-3}	1024	5000

TABLE VIII: Hyperparamter in the W2B experiment.

Model	LR	Batch Size (data)	Batch Size (distributions)	Total Iterations
HOTET	10^{-3}	1024 (dim=2,4,8,16) / 256 (dim=32, 64)	8	5000
MetaOT	10^{-3}	1024 (dim=2,4,8,16) / 256 (dim=32, 64)	8	5000
MMv2	10^{-3}	1024	N/A	5000

TABLE IX: Hyperparamter in the OT maps prediction experiment.

Model	LR	Batch Size (data)	Batch Size (images)	Total Iterations
HOTET (one-to-one)	10^{-3}	1024	N/A	5000
HOTET (multi-to-one)	10^{-3}	1024	8	5000

TABLE X: Hyperparamter in the color transfer experiment.