OT-Transformer: A Continuous-time Transformer Architecture with Optimal Transport Regularization

Kelvin Kan [*] [1] Xingjian Li [*] [2] Stanley Osher [1]

Abstract

Transformers have achieved state-of-the-art performance in numerous tasks. In this paper, we propose a continuous-time formulation of transformers. Specifically, we consider a dynamical system whose governing equation is parametrized by transformer blocks. We leverage optimal transport theory to regularize the training problem, which enhances stability in training and improves generalization of the resulting model. Moreover, we demonstrate in theory that this regularization is necessary as it promotes uniqueness and regularity of solutions. Our model is flexible in that almost any existing transformer architectures can be adopted to construct the dynamical system with only slight modifications to the existing code. We perform extensive numerical experiments on tasks motivated by natural language processing, image classification, and point cloud classification. Our experimental results show that the proposed method improves the performance of its discrete counterpart and outperforms relevant comparing models.

1. Introduction

Transformers were first introduced in (Vaswani et al., 2017) for natural language processing (NLP) tasks. The key feature of the model is the self-attention mechanism, which can capture dependencies of long sequences of data in a parallel manner. This renders the training of transformers more efficient than other architectures, such as RNNs and CNNs, especially when long sequences of data are involved. Since then, not only did it achieve state-of-the-art results in NLP (Radford et al., 2019), but it also found various successful applications, including computer vision (Dosovitskiy et al., 2021), program synthesis (Chen et al., 2021b), computational biology (Jumper et al., 2021), speech processing (Baevski et al., 2020), reinforcement learning (Chen et al., 2021a; Lin et al., 2024), operator learning (Li et al., 2023; Yang et al., 2023) and climate modeling (Gao et al., 2023; Nguyen et al., 2023; 2024).

The basic structure of a transformer architecture is transformer blocks, where self-attention is a key characteristic. In each transformer block, the self-attention layer can capture relationships within the input data in a parallel and efficient manner. The parallel computation of self-attention enhances the transformer's efficiency while preserving its representational power.

Each transformer block also incorporates a skip-connection structure. Inspired by the popular Neural ODE framework (Chen et al., 2018), we propose a continuous-time formulation for transformers, where the hidden states evolve over time according to an ODE. We further leverage optimal transport theory to regularize the hidden state dynamics. We justify this regularization both theoretically and experimentally.

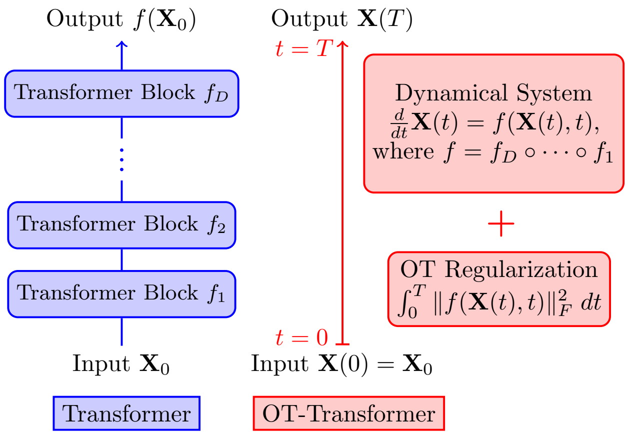

Figure 1. A schematic comparison between the transformer blocks of a **left:** vanilla transformer and **right:** OT-Transformer. OT-Transformer can directly reuse the pre-defined architecture f_i's to parametrize the continuous-time dynamics, which only requires slight modification in the existing program.

[*] Equal contribution [1] Department of Mathematics, University of California, Los Angeles, USA [2] Oden Institute for Computational Engineering and Sciences, University of Texas, Austin, USA. Correspondence to: Kelvin Kan <kelvin.kan@math.ucla.edu>.

We name our model OT-Transformer. Our approach is flexible and straightforward to implement in the sense that one can directly use a predefined transformer architecture to parametrize the ODE. This requires only slight modifications to existing code and opens up possibility for adapting existing architecture. When a single step forward Euler integration scheme is used, our model coincides with the original discrete transformer. Hence, OT-Transformer includes the original transformer as a special case. See Figure 1 for a schematic comparison between the transformer blocks of a pre-defined transformer and OT-Transformer.

We summarize our contributions as follows:

- We propose a continuous-time architecture of transformers. We composite transformer blocks to formulate an ODE governing the dynamics of the hidden states of the transformer. To the best of our knowledge, our approach is distinct from existing transformer models.

- Leveraging optimal transport theory, we use a regularization term penalizing the square arc length of the hidden state trajectory. We remark that the application of optimal transport to the design of transformer architecture remains underexplored and has shown very limited success.

- We demonstrate the effectiveness of the regularization. On the theoretical side, we apply optimal control theory to show that the unregularized training problem is ill-posed, that is, the solution is not unique and hence can be highly irregular. On the empirical side, our experimental results show that the regularization term improves generalization and leads to significantly more numerically stable training across different applications.

- Our experimental results show that our approach improves the performance of the vanilla architecture. In particular, it yields better performance with a reduced number of parameters, which leads to better memory efficiency at inference. In addition, our model outperforms existing continuous-time transformer models.

2. Background

In this section, we discuss the related work that motivate our approach.

Notations In this paper, we use bold uppercase letter (e.g., $\mathbf{X}$) to denote matrices and bold lowercase letter (e.g., $\mathbf{x}$) to denote vectors. Moreover, we use $\mathbf{x}_j$ (resp. $\mathbf{x}_{i,j}$) to represent the jth column of $\mathbf{X}$ (resp. $\mathbf{X}_i$).

Transformers In general, a transformer architecture is formulated as follows. Given an input $\mathbf{Z} = [\mathbf{z}_1, \mathbf{z}_2, ..., \mathbf{z}_n] \in \mathbb{R}^{d_f \times n}$, where n is the number of tokens and d_f is their dimension, it first computes the input embedding of each token by

$$\mathbf{x}_{0,j} = g_l(\mathbf{z}_j; \boldsymbol{\gamma}_l), \quad \text{for} \quad j = 1, 2, ..., n. \tag{1}$$

Here $\mathbf{x}_{0,j} \in \mathbb{R}^d$. The input embedding g_l parametrized by weights $\boldsymbol{\gamma}_l$ embeds each token into a d-dimensional space and incorporates sequential order information into each token. Then, it is processed through a series of transformer blocks, where the output of each block serves as the input to the next. At each step, the model sequentially applies the operation $\mathbf{X}_{i+1} = f_{i+1}(\mathbf{X}_i)$ given by (Thickstun, 2021)[1]

$$\mathbf{u}_{i,j} = \mathbf{x}_{i,j} + \sum_{h=1}^{H} \mathbf{W}_i^h \mathbf{V}_i^h \mathbf{X}_i \operatorname{softmax}\left(\frac{(\mathbf{K}_i^h \mathbf{X}_i)^\top \mathbf{Q}_i^h \mathbf{x}_{i,j}}{\sqrt{k}}\right), \tag{2}$$

$$\mathbf{x}_{i+1,j} = \mathbf{u}_{i,j} + g_f(\mathbf{u}_{i,j}; \boldsymbol{\theta}_i), \tag{3}$$

for $j = 1, 2, ..., n$, and $i = 0, 1, ..., D-1$, where D is the total number of transformer blocks, H is the number of self-attention heads, $\mathbf{Q}_i^h, \mathbf{K}_i^h, \mathbf{V}_i^h \in \mathbb{R}^{k \times d}$ are known as query, key, and value matrices, and $\mathbf{W}_i^h \in \mathbb{R}^{d \times k}$. In (3), a fully connected layer g_f, parametrized by weights $\boldsymbol{\theta}_i$, is applied individually to each of the n tokens. The first equation (2) is known as self-attention layers and is the key feature of transformer architectures. Their matrix multiplication formulation enables the parallel computation of dependencies among tokens, rendering them particularly effective for handling long sequences of tokens, that is, when n is large. This self-attention mechanism enables models to focus on the most relevant parts of an input sequence, adapting dynamically to the context. Its flexibility allows it to capture complex, long-distance relationships within data, different from CNNs which primarily focus on local patterns, and RNNs, which experience a sharp performance decrease with long sequences. Such features make transformers particularly powerful for tasks such as language understanding and image recognition. This series of transformer blocks is also called an encoder in the literature.

Eventually, $\mathbf{X}_D$ is either passed to a decoder comprising another series of transformer blocks and then a multilayer perceptron (MLP) for sequence generation tasks or directly to an MLP for various downstream tasks, including classification and regression. The transformer output $\tilde{\mathbf{y}}$ is therefore computed by

$$\tilde{\mathbf{y}} = g_o(\mathbf{X}_D; \boldsymbol{\gamma}_o), \tag{4}$$

where g_o is either the composition of a decoder and an MLP or an MLP, parametrized by weights $\boldsymbol{\gamma}_o$.

[1] Layer normalization is commonly applied in each transformer block (Xiong et al., 2020). For brevity of exposition, it is omitted in the discussion. But it is included in our experiments.

ResNets and Neural ODEs Residual networks (ResNets) (He et al., 2016) are an extensively employed model which features a skip-connection structure in their layers. Given input $\mathbf{x}_0$, the output of the ith layer is computed by

$$\mathbf{x}_{i+1} = \mathbf{x}_i + g_i(\mathbf{x}_i). \tag{5}$$

Here, g_i is a network layer, and the skip-connection (5) is a key feature of ResNet. This architecture is often compared with the explicit Euler discretization of an ordinary differential equation (ODE) (Weinan, 2017; Haber & Ruthotto, 2017; Ruthotto & Haber, 2020). Based on this insight, (Chen et al., 2018) proposed Neural ODEs (NODEs), whose formulation is given by

$$\frac{d\mathbf{x}(t)}{dt} = f_{\text{NODE}}(\mathbf{x}(t), t). \tag{6}$$

Here $t \in [0, T]$ is artificial time and f_{NODE} is a neural network parametrizing the dynamics. Given an input $\mathbf{x}(0)$, the final output $\mathbf{x}(T)$ is obtained by integrating (6). A notable and relevant advantage of Neural ODEs is their parameter efficiency, as the continuous formulation allows them to model complex transformations over time with fewer parameters compared to traditional architectures.

OT-based CNFs A prominent application of NODEs is continuous normalizing flows (CNFs) (Chen et al., 2018). CNFs use (6) to paramtrize invertible mappings between a standard Gaussian distribution and an unknown target distribution. The ill-posed nature of the CNF formulation can often add to the complexity and computational cost for solving a problem. Optimal transport (OT) based regularization has prominent applications in CNFs and is a powerful tool in improving accuracy and at times reducing cost. Among the infinitely many mappings between the two distributions, OT-based CNFs (Finlay et al., 2020; Yang & Karniadakis, 2020; Onken et al., 2021; Vidal et al., 2023) target to find the optimal transport mapping. This is done by incorporating into the training objective regularization term(s) enforcing straight trajectories in (6). This renders the training problem well-posed (Huang et al., 2023; Zhang & Katsoulakis, 2023). The straight trajectories also offer numerical advantages, as they make the numerical integration of (6) more tractable.

3. OT-Transformers

In this section, we introduce the continuous-time transformer with optimal transport regularization (OT-Transformer). A key feature of OT-Transformer is that, the model uses a combination of transformer blocks and NODE formulation. Specifically, the model parametrizes an ODE using transformer blocks, with the embedded inputs (1) serving as the initial state of the ODE, and the terminal state will be passed to the output layer (4). An optimal transport regularization is used in the training problem, and we demonstrate its benefits empirically and theoretically.

Model Formulation Motivated by the connection between ResNet and neural ODEs, and the inherent skip-connection structure of transformer blocks (2) and (3), we formulate a continuous-time transformer.

Given an input sequence $\mathbf{Z} = [\mathbf{z}_1, \mathbf{z}_2, ..., \mathbf{z}_n]$ of length n, we first apply the input embedding (1) to obtain the initial state $\mathbf{X}(0) = \mathbf{X}_0 \in \mathbb{R}^{d \times n}$. The dynamics of the hidden state is then governed by the ODE[2]

$$\frac{d\mathbf{X}(t)}{dt} = f(\mathbf{X}(t), t; \boldsymbol{\theta}), \quad \text{for} \quad t \in [0, T], \tag{7}$$

where f is the composition of a sequence of transformer blocks defined in (2) and (3), that is, $f = f_D \circ f_{D-1} \circ ... \circ f_1$, and $\boldsymbol{\theta}$ collectively denotes their trainable parameters $\boldsymbol{\theta}_i$, $\mathbf{K}_i^h$, $\mathbf{V}_i^h$, $\mathbf{Q}_i^h$ and $\mathbf{W}_i^h$ for all h and i. In real implementation, we adopt a discretize-then-optimize approach (Onken & Ruthotto, 2020; Onken et al., 2021) and compute the terminal state $\mathbf{X}(T)$ by using numerical integration schemes such as forward Euler or Runge–Kutta methods (Butcher, 2016). Finally, we obtain the transformer output $\tilde{\mathbf{y}}$ by applying (4) to the terminal state $\mathbf{X}(T)$.

Our framework is flexible in that it can be applied to almost any existing transformer architectures. It can directly reuse the architecture of an existing transformer's input embedding, decoder and output layers and use its transformer blocks f_i's to construct the ODE (7)[3]. This only requires slight modifications to existing code. Our framework generalizes the discrete formulation of transformer blocks to continuous-time, effectively enables a continuous-depth formulation of transformer blocks. When $T = 1$ and a single step forward Euler integration scheme is used, our framework is identical to the original discrete transformer formulation. Hence, our framework is consistent with the standard transformer architecture.

Problem Formulation We formulate the training objective as

$$\min_{\boldsymbol{\theta}, \boldsymbol{\gamma}} \mathbb{E} \left\{ L(\mathbf{X}(T), \mathbf{y}; \boldsymbol{\gamma}) + \frac{\lambda}{2dn} \int_0^T \|f(\mathbf{X}(t), t; \boldsymbol{\theta})\|_F^2 \, dt \right\}. \tag{8}$$

[2] We found that including the time variable t as an input yields similar performance to excluding it. Therefore, in our implementation, we do not include the time variable t. That is, the right-hand side of the ODE is $f(\mathbf{X}(t); \theta)$. This simpler option allows us to directly reuse pre-defined transformer block architecture.

[3] It is possible to construct a continuous-time formulation for the decoder, where the decoder is used to formulated a second ODE. In this work, we focus on the continuous-time formulation for the encoder only and leave that for future work.

Here, the expectation is taken over the input-output pairs $(\mathbf{Z}, \mathbf{y})$, $\|\cdot\|_F$ denotes the Frobenius norm, $\boldsymbol{\gamma}$ collectively denotes the weights of the input embedding $\boldsymbol{\gamma}_l$ and output layer $\boldsymbol{\gamma}_o$, $\boldsymbol{\theta}$ collectively denotes the weights of the transformer blocks $\mathbf{W}_i^h$, $\mathbf{Q}_i^h$, $\mathbf{K}_i^h$, $\mathbf{V}_i^h$, and $\boldsymbol{\theta}_i$'s. The loss function L measures the difference between the target output $\mathbf{y}$ and model output $\tilde{\mathbf{y}}(\mathbf{X}(\mathbf{T}); \boldsymbol{\gamma})$. For instance, in classification (Dosovitskiy et al., 2021) and sequence generation (Vaswani et al., 2017) tasks, one commonly uses the softmax loss. The second term is a transport cost regularization penalizing the squared norm of the velocity (the right hand side of the ODE). It enhances the regularity of the hidden state dynamics (7) by promoting more constant speed and straighter state trajectories. In practice, this regularization term is computed easily as it is calculated alongside the numerical integration of the ODE (7). The regularization term is normalized by $1/dn$, where dn is the dimension of f. This normalization accounts for the size of f, ensuring the regularization term remains consistent across different dimensions. The regularization parameter λ balances the effects of the two terms.

Empirical Benefits of Transport Cost As we will demonstrate empirically in our experiments, the transport cost improves the model's effectiveness. The transport cost serves as a regularizer and can stabilize the training process; without it, the model is more prone to experiencing exploding or vanishing gradients. Moreover, the generalization (i.e., performance on unseen data) of the model is enhanced, thanks to the more regular hidden state dynamics. Interestingly, the regularized model can also achieve a lower data-fitting loss for the training data, despite the incorporation of the regularization term. This occurs because the optimization process is stochastic, with each model update based on a batch of data rather than the entire training set. The regularization term, however, encourages broader generalization across the entire dataset.

Theoretical Benefits of Transport Cost We theoretically demonstrate the purpose of the transport cost. Specifically, using optimal control theory (Kirk, 2004; Liberzon, 2011), we show that the training problem is ill-posed without the transport cost regularization. In particular, the solution is not unique and thus can be highly irregular. We build upon the anaylses of (Zhang & Katsoulakis, 2023; Gu et al., 2024), which study OT-based CNFs for learning a marginal distribution. We modify their approach to adapt to the case where the solution is conditional on $\mathbf{y}$.

For a given target output $\mathbf{y}$, optimal control theory (Fleming & Rishel, 2012; Liberzon, 2011) states that there exists a potential function $\Phi_{\mathbf{y}} : \mathbb{R}^{d\times n} \times [0, T] \to \mathbb{R}$, where the optimal f for (8) can be represented by

$$f(\mathbf{X}, t) = -\frac{dn}{\lambda} \nabla \Phi_{\mathbf{y}}(\mathbf{X}, t), \tag{9}$$

where the gradient $\nabla \Phi_{\mathbf{y}}(\mathbf{X}, t)$ is taken with respect to the first argument $\mathbf{X}$. This is analogous to classical physics, where $\mathbf{X}$ moves in a manner to minimize its potential. Optimal control theory further states that the Hamilton-Jacobi-Bellman (HJB) equation (Bellman, 1954; Evans, 2010) is an optimality condition characterizing the optimal value $\Phi_{\mathbf{y}}$ and is given by

$$\begin{aligned} -\partial_t \Phi_{\mathbf{y}}(\mathbf{X}, t) + H(\mathbf{X}, \nabla \Phi_{\mathbf{y}}(\mathbf{X}, t)) = 0, \\ \Phi_{\mathbf{y}}(\mathbf{X}, T) = L(\mathbf{X}, \mathbf{y}), \end{aligned} \tag{10}$$

where the Hamiltonian $H : \mathbb{R}^{d\times n} \times \mathbb{R}^{d\times n} \to \mathbb{R}$ is given by

$$H(\mathbf{X}, \mathbf{P}) = \sup_f -\langle \mathbf{P}, f(\mathbf{X}, t)\rangle - \frac{\lambda}{2dn} \|f(\mathbf{X}, t)\|_F^2, \tag{11}$$

with $\langle \cdot, \cdot \rangle$ representing the Frobenius inner product, $\mathbf{P}$ is the adjoint variable to the system and is introduced by the Pontryagin Maximum Principle (Mangasarian, 1966; Fleming & Rishel, 1975). We see that when $\lambda = 0$, i.e., when the transport cost is absent, the Hamiltonian cannot be defined properly and equals infinity. Therefore, there is no well-defined HJB equation, and the training problem (8) becomes degenerate. As such there are infinitely many choices of f that minimize the data fidelity term in (8), including some highly irregular ones. For instance, f can produce a zig-zagging hidden state trajectory or move to the target location instantly and then remain stationary. These irregular paths can pose challenges in numerical integration and result in numerical instability during training, as demonstrated by our experiments. On the other hand, the addition of the transport cost promotes the uniqueness and regularity of the solution. In short, the training problem is well-posed only if the corresponding HJB equation is well-posed (Lasry & Lions, 2007; Bensoussan et al., 2013).

4. Related Work

This section provides a review of relevant work.

Continuous-time Architecture There has been some works on a continuous-time interpretation of transformers. And there is a key distinction between the formulations of OT-Transformer and existing models. In OT-Transformer, we use the composition of all transformer blocks to parametrize a single dynamical system (7) governing the hidden states. To the best of our knowledge, the existing works use each transformer block to parametrize a dynamical system. For a transformer with D transformer blocks, the continuous-time model is represented as the output of D different dynamical systems. In particular, it is formulated as

$$\begin{aligned} &\mathbf{X}_0(0) = \mathbf{X}_0, \\ &\mathbf{X}_i(0) = \mathbf{X}_{i-1}(T), && \text{for } 1 \le i \le D-1, \\ &\frac{d\mathbf{X}_i(t)}{dt} = \hat{f}_i(\mathbf{X}_i(t), t; \hat{\boldsymbol{\theta}}_i), && \text{for } t \in [0,T],\ 0 \le i \le D-1, \end{aligned} \tag{12}$$

where $\hat{f}_i$ is the ith transformer block parametrized by weights $\hat{\boldsymbol{\theta}}_i$ and defined in (2) and (3), except that the fully-connected layer (3) has no skip-connection.

This formulation is introduced in (Baier-Reinio & De Sterck, 2020), the model of which is conceptually the closest to our approach. Here, we highlight several key differences between their work and ours. Firstly, they only conduct the simple task of determining the parity of a binary sequence, rather than investigating its performance in general applications. When their approach is applied, it fails to improve performance over the vanilla transformer and instead degrades it. While they also propose the use of the transport cost, the regularization cannot improve the performance of their model when the sequence length exceeds eight. We observe similar issues when testing their model on other applications; see Section 5. This is potentially due to their choice of formulation. Specifically, in (12), as the model transitions from one transformer block to the next, it effectively switches to a different dynamical system, introducing non-smoothness to the overall dynamics. This undermines the purpose of the transport cost regularization, which seeks to obtain a continuous and more constant velocity. In contrast, our model is formulated using only one dynamical system. The resulting dynamics is smoother and thus inherently better suited to incorporate the transport cost regularization. This is evident in our experimental results, while the regularization can always improve the generalization of OT-Transformer to a significant extent, this is not the case with their model; in certain scenarios, the regularization may even degrade their model's performance. Moreover, they do not provide theoretical analysis to the regularization or demonstrate its numerical advantages. We also mention that, while (Baier-Reinio & De Sterck, 2020) proposes alternative formulations for further investigation, it does not consider ours, highlighting the novelty and non-triviality of our approach.

Since then, there have been a number of follow-up works that build on the formulation Equation (12) to perform different tasks, including sequence generation (Lu et al.; Li et al., 2021; 2022; Zhong et al., 2022), time series forecasting (Xu et al., 2023; Cheng et al., 2024), and image classification (Niu et al., 2024; Okubo et al., 2024). But most of these methods only use the formulation (12) as motivation and are discrete architectures in nature, and none of them consider transport cost regularization in their approach. Moreover, these models focused on a specific type of application and not general-purpose.

In order to access the performance of our OT-Transformer more comprehensively, we also include the existing transformer formulation (12) as a benchmark in our experiments. It is referred to as "N-ODE Transformer" in our experimental results, following the terminology in (Baier-Reinio & De Sterck, 2020).

Mathematical Analysis There have been works that theoretically analyze a continuous-time formulation of transformers. In (Geshkovski et al., 2023; 2024a), they show that a continuous-time formulation can be interpreted as an interacting particle system, where each token can be perceived as a particle. They demonstrate that there is a clustering behavior among the tokens. Since then, there has been a number of works that further investigate the dynamics of tokens through this interpretation, including (Adu & Gharesifard, 2024; Bruno et al., 2024; Biswal et al., 2024; Geshkovski et al., 2024b; Karagodin et al., 2024), to name a few. However, we note that the aforementioned work is primarily theoretical and lacks evaluations beyond toy experiments. In (Sander et al., 2022), they show that, under some restriction on the weights, a continuous-time formulation of self-attention layers can be interpreted as a gradient flow. However, no experiments have been conducted following this analysis.

5. Experimental Results

We demonstrate the advantage of our proposed OT-transformers through four extensive experiments arising from point cloud classification, image classification, and text sentiment analysis.

For each task, we use commonly used transformer architectures as baselines. All the hyperparameters of the experiments, including architectures of baseline models, number of epochs, learning rates, layer normalization, etc., are identical to those used in (Sander et al., 2022). We also compared against N-ODE Transformer, an existing continuous-time transformer formulation which is introduced in (Baier-Reinio & De Sterck, 2020) and has been considered in other works. For details about the formulation and specific applications, see the discussion in Section 4. In the reported results, we refer to N-ODE Transformer with and without transport cost as unregularized N-ODE Transformer and regularized N-ODE Transformer, respectively.

For the continuous-time models, we employ the same architectures as the baselines but with a reduced hidden dimensions or number of layers for the transformer blocks. This is for investigating their parameter efficiency. To demonstrate the effectiveness of the transport cost on OT-Transformer, we also perform the experiments with $\lambda = 0$ in (8), effectively creating an unregularized model. We label this model unregularized OT-transformer in the reported results. For the continous-time models, we use an explicit Euler scheme to numerically integrate the dynamical systems.

For more details of the experiments, we refer our readers to Appendix B. Our program is implemented using PyTorch (Paszke et al., 2017) and executed using NVIDIA A100 GPUs.

5.1. Point Cloud Classification

We use the ModelNet 40 dataset (Wu et al., 2015), which is among the most widely used benchmark for point cloud classification (Uy et al., 2019). The dataset contains roughly 10,000 Computer-Aided Design (CAD) models that are categorized into 40 distinct classes, including common objects such as airplanes, cars, and furniture.

We experiment with the Set Transformer model (Lee et al., 2019). It has an encoder-decoder architecture and is specifically designed to process unordered data, such as point clouds, ensuring that the output remains permutation invariant to its input. Following the setup of (Sander et al., 2022), we use the baseline architecture with two Induced Self Attention Blocks (ISABs) (Lee et al., 2019) in the encoder, where each ISAB contains two transformer blocks, and experiment with 5,000 uniformly sampled points for each shape. For the continuous-time models, we use the same architecture except that we put a fully-connected layer before the transformer blocks so that the dimension is consistent for continuous-time dynamics. Also the hidden dimensions d and k of the ISABs are reduced from 256 to 200. This reduces the number of parameters for the ISABs by 24%.

We perform the experiment over five random trials and report the best test accuracies in Table 1. The unregularized continuous-time models encountered gradient explosion, resulting in NaN outputs, and the issue persists even with slight regularization. We found that the models never suffered from gradient explosion with sufficient regularization, indicating that transport cost effectively stabilizes the training process. Hence, we only report the performance of the regularized models. The baseline Set Transformer obtains an average test accuracy of 87.4%. The regularized N-ODE Transformer achieves an accuracy of 87.5%, indicating negligible improvement over the vanilla model. Our OT-Transformer shows a sizable improvement and reports an average 89.9% test accuracy even with a smaller model. From the learning curves in Figure 2, we see that our model reports a lower data-fitting loss for training data compared to the vanilla model, despite the inclusion of a regularization term,

Table 1. Number of parameters for the transformer blocks, mean test accuracy and standard deviation (std) over five trials for the point cloud experiment.

Method/Exp.	Para. Count	Test Accuracy
Baseline	0.86M	$87.4\% \pm 0.45\%$
Reg. N-ODE Trans.	0.65M	$87.5\% \pm 0.51\%$
OT-Trans. (Ours)	0.65M	$\mathbf{89.9\% \pm 0.42\%}$

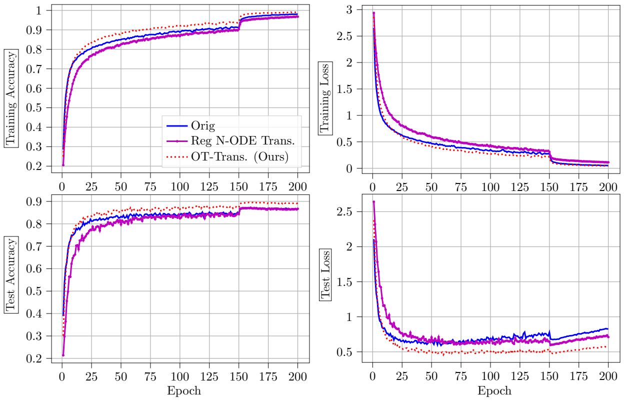

Figure 2. Accuracy and data-fitting loss for the point cloud experiment (averaged over five trials)

5.2. Image Classification

To further demonstrate the applicability of our proposed method, we also perform experiments on imaging tasks. We consider the Vision Transformer (ViT), which was introduced in (Dosovitskiy et al., 2021). Since then, the model and its variants have achieved state-of-the-art performance in computer vision tasks (Ruan et al., 2022; Xia et al., 2024). The key feature of ViTs is that they divide an image into fixed-size patches, which are treated as sequences of data. ViTs then apply self-attention mechanisms to capture relationships between these patches, enabling it to learn complex structures across the entire image. We perform two image classification experiments following the same setup as in (Sander et al., 2022).

MNIST Classification We first conduct a small-scale image classification experiment with the MNIST dataset (LeCun, 1998). Following (Sander et al., 2022), the baseline model ViT has one transformer block with a single-head self-attention layer and no fully-connected layer. Since it has only one transformer block, N-ODE Transformer and our OT-Transformer share the same formulation, and we report the results as OT-Transformer.

The OT-Transformer uses the same model architecture as the baseline model, except that the hidden dimensions d and k of the self-attention layer are reduced to 64 from 128.

This reduces the number of parameters by over 80%. The experiments are conducted over five random trials. The best test accuracies are reported in Table 2. OT-Transformer demonstrates significant improvements over the baseline in both accuracy and model efficiency. The baseline model achieved a test accuracy of 93.0%. The unregularized OT-Transformer improves the test accuracy to 96.8%, although it uses a much smaller model architecture. The transport cost regularization further improves the test accuracy to 97.1% while maintaining the same reduced parameter count. Notably, OT-Transformer also exhibits significantly lower standard deviation across five trials when compared to the baseline and unregularized model, indicating enhanced stability and reliability in its performance. Interestingly, when we compare the learning curves of the unregularized and regularized OT-Transformers in Figure 3, we observe that including the transport cost regularization also reduces the training loss for data-fitting and accuracy.

Table 2. Number of parameters for different models, mean test accuracy and standard deviation over five trials for the MNIST image classification experiment.

Method/Exp.	**Para. Count**	**Test Accuracy**
Baseline	93k	$93.0\% \pm 0.69\%$
Unreg. OT-Trans.	18k	$96.8\% \pm 0.23\%$
OT-Trans. (Ours)	18k	$\mathbf{97.1\% \pm 0.16\%}$

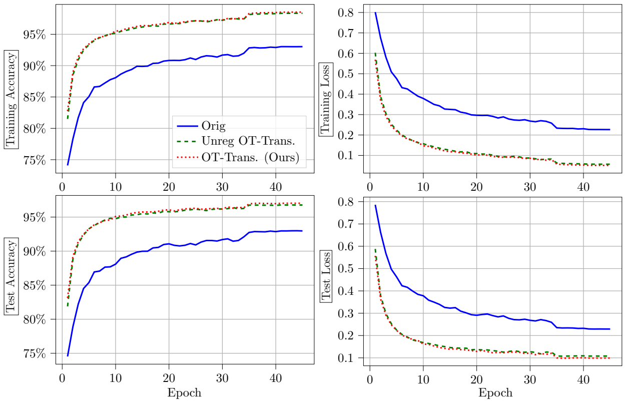

Figure 3. Accuracy and data-fitting loss for the MNIST image classification experiment (averaged over five trials)

Cats and Dogs Classification We perform experiments on a binary cats and dogs image classification task, following (Sander et al., 2022). The baseline ViT has six layers of transformer blocks. We choose the continuous-time counterparts to have five layers; this reduces the number of parameters for the transformer blocks by around 20%. We report in Table 3 the test accuracies after the last epoch, which demonstrate a more significant improvement. The best test accuracy is also reported in Appendix B, where our model also performs best. We observe again that our OT-Transformer has the best performance and obtains a test accuracy of 79.0%, improving from the baseline's 77.6%. The standard deviation of the test accuracy, at 0.31%, is significantly lower than the baseline value of 0.86%, showing our proposed approach is more robust and reliable. We also observe that incorporating the transport cost regularization improves generalization and stability of OT-Transformer; without it, the average and standard deviation of test accuracy worsen to 78.2% and 0.39%, respectively. Both the unregularized and regularized N-ODE Transformers report a test accuracy of 75.6%, which is worse than the baseline model, making them undesirable methods for the problem. Unlike our model, incorporating the regularization also has little effect on the performance of N-ODE Transformer. This is likely due to the incompatibility of N-ODE Transformer and the regularization; see Section 4. We report the learning curves in Figure 4. When we compare the learning curves of the unregularized and regularized OT-Transformers, we see that including the transport cost regularization also improves the training loss for data-fitting and accuracy.

Table 3. Number of parameters for the transformer blocks, mean test accuracy after the last epoch and standard deviation over three trials for the cats and dogs image classification experiment.

Method/Exp.	**Para. Count**	**Test Accuracy**
Baseline	1.77M	$77.6\% \pm 0.86\%$
Unreg. N-ODE Trans.	1.48M	$75.6\% \pm 0.48\%$
Reg. N-ODE Trans.	1.48M	$75.6\% \pm 0.03\%$
Unreg. OT-Trans.	1.48M	$78.2\% \pm 0.39\%$
OT-Trans. (Ours)	1.48M	$\mathbf{79.0\% \pm 0.31\%}$

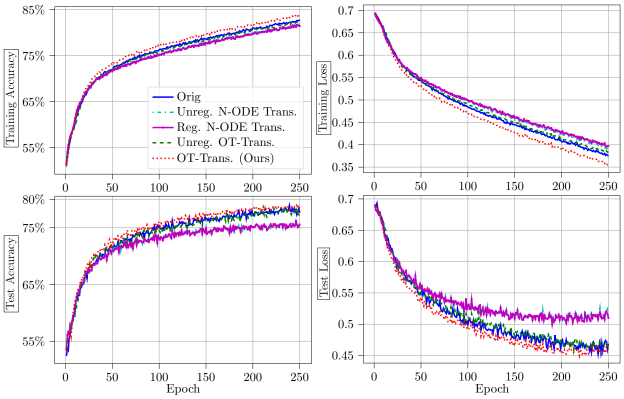

Figure 4. Accuracy and data-fitting loss for the cats and dogs image classification experiment

5.3. Sentiment Analysis

We perform sentiment analysis on the IMDb movie review dataset (Maas et al., 2011), aiming to predict whether each movie review is positive or negative. We use an identical baseline transformer architecture as in (Sander et al., 2022), which has six layers of transformer blocks. The OT-Transformer counterpart has only 3 layers, reducing the number of parameters of the transformer blocks by half.

We repeat the experiment for five random trials. In all trials, the unregularized N-ODE Transformer and OT-Transformer experienced issues with exploding gradients, resulting in NaN outputs. In order to estimate how the unregularized model would perform under more stable conditions, we impose a slight transport cost with $\lambda = 0.01$. We note that the continuous-time models with slight and standard regularization completed all trials without issues. This shows the effectiveness of the transport cost regularization in stabilizing the training process and avoiding exploding gradients.

The best test accuracies are reported in Table 4. The baseline architecture achieved a test accuracy of 83.9%. The N-ODE Transformers with slight and standard regularization report a test accuracy of 83.6% and 83.9%, respectively, which are not better than the baseline model. The N-ODE Transformer with slight regularization reports a test accuracy of 83.6%. With a standard regularization, the test accuracy slightly increases to 83.9%. However, both results are not better than that of the baseline model. The OT-Transformer with slight regularization reported a test accuracy of 82.7%, which is subpar compared to the baseline model. On the other hand, the standard OT-Transformer achieves the best test accuracy of 84.6%, which is 0.7% higher than the baseline model, in spite of using a smaller model. The test accuracy is also 0.7% higher than that of the N-ODE Transformer's. We note that with the incorporation of transport cost, the accuracy of N-ODE Transformer is improved by only 0.3%. In contrast, the accuracy of OT-Transformer is boosted by 1.9%. Again, this is likely due to that our continuous-time formulation is inherently more suited for transport cost regularization than that of N-ODE Transformer; see Section 4 for the more detailed discussion.

The learning curves are reported in Figure 5. When we compare the results of the unregularized and regularized OT-Transformers, we see that the regularization effectively reduces overfitting by increasing training loss while simultaneously lowering test loss. Overall, we see that the combination of our continuous-in-time formulation and transport cost regularization enhances parameter efficiency and generalization of transformers.

Table 4. Mean test accuracy and standard deviation over five trials for the sentiment analysis experiment. *: The unregularized continuous-time models experienced gradient explosion. And we estimate their performance by using a slight regularization $\lambda = 0.01$.

Method/Exp.	Para. Count	Test Accuracy
Baseline	4.74M	$83.9\% \pm 0.26\%$
Unreg. N-ODE Trans.	2.37M	$83.6\% \pm 0.40\%^*$
Reg. N-ODE Trans.	2.37M	$83.9\% \pm 0.48\%$
Unreg. OT-Trans.	2.37M	$82.7\% \pm 0.38\%^*$
OT-Trans. (Ours)	2.37M	$\mathbf{84.6\% \pm 0.55\%}$

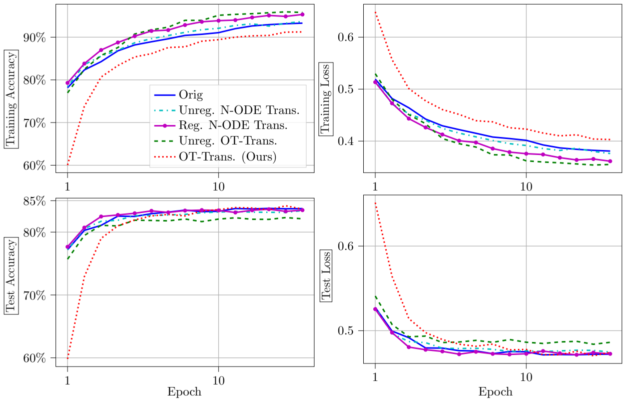

Figure 5. Accuracy and data-fitting loss for the sentiment analysis experiment

6. Discussion and Summary

We proposed OT-Transformer, a continuous-time formulation of transformers. OT-Transformer is flexible and is general-purpose, as it can be easily adapted to different variations of the vanilla transformer architecture, making it suitable for a wide class of tasks. It is also distinctive from existing continuous-time transformer architecture. Our training objective includes a transport cost regularization, which we justified through theory and extensive experimentation. In particular, we showed that the training problem is ill-posed without the regularization. We also illustrated that the regularization stabilizes the training process and enhances the generalization of our model. Through multiple tests across different applications, we demonstrate that our model improves the baseline transformer architecture in terms of parameter efficiency and accuracy, while reducing the variance among trials at the same time. This is particularly beneficial during inference; without the need for gradient tracking, our smaller models are more memory efficient. Contributing to the point, we also notice that it is possible to reduce the number of time steps at the cost of minor decrease in performance during inference, see Appendix A. Most importantly, it outperforms the existing continuous-time transformer architecture. These results showcase the effectiveness and potential of our model.

Impact Statement

This paper presents work whose goal is to advance the field of Machine Learning. There are many potential societal consequences of our work, none of which we feel must be specifically highlighted here.

Acknowledgements

The authors would like to thank Lars Ruthotto and Tingwei Meng for the valuable advice and insightful discussion.

References

Adu, D. O. and Gharesifard, B. Approximate controllability of continuity equation of transformers. *IEEE Control Systems Letters*, 2024.

Baevski, A., Zhou, Y., Mohamed, A., and Auli, M. wav2vec 2.0: A framework for self-supervised learning of speech representations. *Advances in neural information processing systems*, 33:12449–12460, 2020.

Baier-Reinio, A. and De Sterck, H. N-ODE transformer: A depth-adaptive variant of the transformer using neural ordinary differential equations. *arXiv preprint arXiv:2010.11358*, 2020.

Bellman, R. Dynamic programming and a new formalism in the calculus of variations. *Proceedings of the national academy of sciences*, 40(4):231–235, 1954.

Bensoussan, A., Frehse, J., Yam, P., et al. *Mean field games and mean field type control theory*, volume 101. Springer, 2013.

Biswal, S., Elamvazhuthi, K., and Sonthalia, R. Identification of mean-field dynamics using transformers. *arXiv preprint arXiv:2410.16295*, 2024.

Bruno, G., Pasqualotto, F., and Agazzi, A. Emergence of meta-stable clustering in mean-field transformer models. *arXiv preprint arXiv:2410.23228*, 2024.

Butcher, J. C. *Numerical methods for ordinary differential equations*. John Wiley & Sons, 2016.

Chen, L., Lu, K., Rajeswaran, A., Lee, K., Grover, A., Laskin, M., Abbeel, P., Srinivas, A., and Mordatch, I. Decision transformer: Reinforcement learning via sequence modeling. *Advances in neural information processing systems*, 34:15084–15097, 2021a.

Chen, M., Tworek, J., Jun, H., Yuan, Q., Pinto, H. P. D. O., Kaplan, J., Edwards, H., Burda, Y., Joseph, N., Brockman, G., et al. Evaluating large language models trained on code. *arXiv preprint arXiv:2107.03374*, 2021b.

Chen, R. T., Rubanova, Y., Bettencourt, J., and Duvenaud, D. K. Neural ordinary differential equations. *Advances in neural information processing systems*, 31, 2018.

Cheng, Y., Li, R., Cheng, J., and Kong, M. Rktrans: Transformer with improved residual connection units for power forecasting. In *2024 2nd International Conference on Pattern Recognition, Machine Vision and Intelligent Algorithms (PRMVIA)*, pp. 54–58. IEEE, 2024.

Dosovitskiy, A., Beyer, L., Kolesnikov, A., Weissenborn, D., Zhai, X., Unterthiner, T., Dehghani, M., Minderer, M., Heigold, G., Gelly, S., Uszkoreit, J., and Houlsby, N. An image is worth 16x16 words: Transformers for image recognition at scale. In *International Conference on Learning Representations*, 2021. URL https://openreview.net/forum?id=YicbFdNTTy.

Evans, L. C. *Partial Differential Equations*, volume 19. American Mathematical Soc., 2010.

Finlay, C., Jacobsen, J.-H., Nurbekyan, L., and Oberman, A. How to train your neural ODE: the world of Jacobian and kinetic regularization. In *International conference on machine learning*, pp. 3154–3164. PMLR, 2020.

Fleming, W. H. and Rishel, R. W. *Deterministic and Stochastic Optimal Control*, volume 1. Springer-Verlag, New York, Heidelberg, Berlin, 1975.

Fleming, W. H. and Rishel, R. W. *Deterministic and stochastic optimal control*, volume 1. Springer Science & Business Media, 2012.

Gao, Z., Shi, X., Wang, H., Zhu, Y., Wang, Y., Li, M., and Yeung, D.-Y. Earthformer: Exploring space-time transformers for earth system forecasting, 2023. URL https://arxiv.org/abs/2207.05833.

Geshkovski, B., Letrouit, C., Polyanskiy, Y., and Rigollet, P. A mathematical perspective on transformers. *arXiv preprint arXiv:2312.10794*, 2023.

Geshkovski, B., Letrouit, C., Polyanskiy, Y., and Rigollet, P. The emergence of clusters in self-attention dynamics. *Advances in Neural Information Processing Systems*, 36, 2024a.

Geshkovski, B., Rigollet, P., and Ruiz-Balet, D. Measure-to-measure interpolation using transformers. *arXiv preprint arXiv:2411.04551*, 2024b.

Gu, H., Katsoulakis, M. A., Rey-Bellet, L., and Zhang, B. J. Combining Wasserstein-1 and Wasserstein-2 proximals: robust manifold learning via well-posed generative flows. *arXiv preprint arXiv:2407.11901*, 2024.

Haber, E. and Ruthotto, L. Stable architectures for deep neural networks. *Inverse problems*, 34(1):014004, 2017.

He, K., Zhang, X., Ren, S., and Sun, J. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 770–778, 2016.

Huang, H., Yu, J., Chen, J., and Lai, R. Bridging mean-field games and normalizing flows with trajectory regularization. *Journal of Computational Physics*, 487:112155, 2023.

Jumper, J., Evans, R., Pritzel, A., Green, T., Figurnov, M., Ronneberger, O., Tunyasuvunakool, K., Bates, R., Žídek, A., Potapenko, A., et al. Highly accurate protein structure prediction with alphafold. *nature*, 596(7873):583–589, 2021.

Karagodin, N., Polyanskiy, Y., and Rigollet, P. Clustering in causal attention masking. *arXiv preprint arXiv:2411.04990*, 2024.

Kirk, D. E. *Optimal control theory: an introduction*. Courier Corporation, 2004.

Lasry, J.-M. and Lions, P.-L. Mean field games. *Japanese journal of mathematics*, 2(1):229–260, 2007.

LeCun, Y. The MNIST database of handwritten digits. *http://yann. lecun. com/exdb/mnist/*, 1998.

Lee, J., Lee, Y., Kim, J., Kosiorek, A., Choi, S., and Teh, Y. W. Set transformer: A framework for attention-based permutation-invariant neural networks. In *International conference on machine learning*, pp. 3744–3753. PMLR, 2019.

Li, B., Du, Q., Zhou, T., Zhou, S., Zeng, X., Xiao, T., and Zhu, J. ODE transformer: An ordinary differential equation-inspired model for neural machine translation. *arXiv preprint arXiv:2104.02308*, 2021.

Li, B., Du, Q., Zhou, T., Jing, Y., Zhou, S., Zeng, X., Xiao, T., Zhu, J., Liu, X., and Zhang, M. ODE transformer: An ordinary differential equation-inspired model for sequence generation. In Muresan, S., Nakov, P., and Villavicencio, A. (eds.), *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 8335–8351, Dublin, Ireland, May 2022. Association for Computational Linguistics. doi: 10.18653/v1/2022.acl-long.571. URL https://aclanthology.org/2022.acl-long.571/.

Li, Z., Meidani, K., and Farimani, A. B. Transformer for partial differential equations' operator learning. *Transactions on Machine Learning Research*, 2023. ISSN 2835-8856. URL https://openreview.net/forum?id=EPPqt3uERT.

Liberzon, D. *Calculus of variations and optimal control theory: a concise introduction*. Princeton university press, 2011.

Lin, L., Bai, Y., and Mei, S. Transformers as decision makers: Provable in-context reinforcement learning via supervised pretraining, 2024. URL https://arxiv.org/abs/2310.08566.

Lu, Y., Li, Z., He, D., Sun, Z., Dong, B., Qin, T., Wang, L., and Liu, T.-y. Understanding and improving transformer from a multi-particle dynamic system point of view. In *ICLR 2020 Workshop on Integration of Deep Neural Models and Differential Equations*.

Maas, A., Daly, R. E., Pham, P. T., Huang, D., Ng, A. Y., and Potts, C. Learning word vectors for sentiment analysis. In *Proceedings of the 49th annual meeting of the association for computational linguistics: Human language technologies*, pp. 142–150, 2011.

Mangasarian, O. L. Sufficient conditions for the optimal control of nonlinear systems. *SIAM Journal on control*, 4 (1):139–152, 1966.

Nguyen, T., Brandstetter, J., Kapoor, A., Gupta, J. K., and Grover, A. Climax: A foundation model for weather and climate, 2023. URL https://arxiv.org/abs/2301.10343.

Nguyen, T., Shah, R., Bansal, H., Arcomano, T., Maulik, R., Kotamarthi, V., Foster, I., Madireddy, S., and Grover, A. Scaling transformer neural networks for skillful and reliable medium-range weather forecasting, 2024. URL https://arxiv.org/abs/2312.03876.

Niu, H., Luo, F., Yuan, B., Zhang, Y., and Wang, J. Efficient visual transformer transferring from neural ODE perspective. *Electronics Letters*, 60(17):e70015, 2024.

Okubo, I., Sugiura, K., and Matsutani, H. A cost-efficient FPGA-based CNN-transformer using neural ODE. *IEEE Access*, 2024.

Onken, D. and Ruthotto, L. Discretize-optimize vs. optimize-discretize for time-series regression and continuous normalizing flows. *arXiv preprint arXiv:2005.13420*, 2020.

Onken, D., Fung, S. W., Li, X., and Ruthotto, L. OT-flow: Fast and accurate continuous normalizing flows via optimal transport. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 35, pp. 9223–9232, 2021.

Paszke, A., Gross, S., Chintala, S., Chanan, G., Yang, E., DeVito, Z., Lin, Z., Desmaison, A., Antiga, L., and Lerer, A. Automatic differentiation in PyTorch. 2017.

Radford, A., Wu, J., Child, R., Luan, D., Amodei, D., Sutskever, I., et al. Language models are unsupervised multitask learners. *OpenAI blog*, 1(8):9, 2019.

Ruan, B.-K., Shuai, H.-H., and Cheng, W.-H. Vision transformers: state of the art and research challenges. *arXiv preprint arXiv:2207.03041*, 2022.

Ruthotto, L. and Haber, E. Deep neural networks motivated by partial differential equations. *Journal of Mathematical Imaging and Vision*, 62(3):352–364, 2020.

Sander, M. E., Ablin, P., Blondel, M., and Peyré, G. Sinkformers: Transformers with doubly stochastic attention. In *International Conference on Artificial Intelligence and Statistics*, pp. 3515–3530. PMLR, 2022.

Thickstun, J. The transformer model in equations. *University of Washington: Seattle, WA, USA*, 2021.

Uy, M. A., Pham, Q.-H., Hua, B.-S., Nguyen, T., and Yeung, S.-K. Revisiting point cloud classification: A new benchmark dataset and classification model on real-world data. In *Proceedings of the IEEE/CVF international conference on computer vision*, pp. 1588–1597, 2019.

Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., Kaiser, L. u., and Polosukhin, I. Attention is all you need. In Guyon, I., Luxburg, U. V., Bengio, S., Wallach, H., Fergus, R., Vishwanathan, S., and Garnett, R. (eds.), *Advances in Neural Information Processing Systems*, volume 30. Curran Associates, Inc., 2017. URL https://proceedings.neurips.cc/paper_files/paper/2017/file/3f5ee243547dee91fbd053c1c4a845aa-Paper.pdf.

Vidal, A., Wu Fung, S., Tenorio, L., Osher, S., and Nurbekyan, L. Taming hyperparameter tuning in continuous normalizing flows using the JKO scheme. *Scientific reports*, 13(1):4501, 2023.

Weinan, E. A proposal on machine learning via dynamical systems. *Communications in Mathematics and Statistics*, 1(5):1–11, 2017.

Wu, Z., Song, S., Khosla, A., Yu, F., Zhang, L., Tang, X., and Xiao, J. 3D ShapeNets: A deep representation for volumetric shapes. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 1912–1920, 2015.

Xia, C., Wang, X., Lv, F., Hao, X., and Shi, Y. Vit-comer: Vision transformer with convolutional multi-scale feature interaction for dense predictions. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 5493–5502, 2024.

Xiong, R., Yang, Y., He, D., Zheng, K., Zheng, S., Xing, C., Zhang, H., Lan, Y., Wang, L., and Liu, T. On layer normalization in the transformer architecture. In *International Conference on Machine Learning*, pp. 10524–10533. PMLR, 2020.

Xu, X., Wang, Z., Zhou, F., Huang, Y., Zhong, T., and Trajcevski, G. Dynamic transformer ODEs for large-scale reservoir inflow forecasting. *Knowledge-Based Systems*, 276:110737, 2023.

Yang, L. and Karniadakis, G. E. Potential flow generator with l 2 optimal transport regularity for generative models. *IEEE Transactions on Neural Networks and Learning Systems*, 33(2):528–538, 2020.

Yang, L., Liu, S., Meng, T., and Osher, S. J. In-context operator learning with data prompts for differential equation problems. *Proceedings of the National Academy of Sciences*, 120(39):e2310142120, 2023.

Zhang, B. J. and Katsoulakis, M. A. A mean-field games laboratory for generative modeling. *arXiv preprint arXiv:2304.13534*, 2023.

Zhong, Y. D., Zhang, T., Chakraborty, A., and Dey, B. A neural ODE interpretation of transformer layers. In *The Symbiosis of Deep Learning and Differential Equations II*, 2022. URL https://openreview.net/forum?id=nA9hvYMQCy.

A. Different Time Stepping at Inference

In this section, we briefly discuss the numerical results of selecting different time step sizes at inference and the changes in model performance. It is evident in the Neural ODE literature, such as (Chen et al., 2018; Onken & Ruthotto, 2020) that using smaller step sizes in time discretization improves integration accuracy and can enhance overall model performance. However, in (Onken et al., 2021) it is also pointed out that such requirement can be relaxed if the underlying dynamics is sufficiently regular, particularly at inference. We here use the aforementioned MNIST experiment in Section 5 to verify the performance change when using different time step sizes to evaluate a pretrained model. Note that the models reported in Section 5 are trained using 20 time steps from time 0 to $T = 1$.

Table 5. Results for the MNIST example, tested using different number of time steps over trained models, here we use accuracy on the test data set to indicate performance.

number of time steps method	1	2	4	8	16	20
Unregularized OT-Transformer	18.2%	42.0%	85.1%	96.3%	96.7%	96.8%
OT-Transformer	17.9%	46.8%	87.1%	96.6%	97.0%	97.1%

We display the results in Table 5. Here, we test different number of time steps from 1 to 20 for both the unregularized model and the OT-Transformer model. Notice here first regularized models performance consistently better than that of the unregularized model, indicating the importance of OT regularization. More importantly we find that it is possible to reduce the number of time steps in evaluation with little decrease in model performance. Specifically we note that decreasing the number of time steps from 20 to 8 only resulted in about 0.5% decrease in test accuracy. We believe this finding can be meaningful, as it suggests further efficiency improvement at the model deployment stage. However, additional testing may be required for other examples, we will leave further investigation of this point for future work.

B. Experimental Details and Results

We report the detailed experimental setups here. We adapted the code provided by (Sander et al., 2022), maintaining the same default data processing setup, hyperparameters, and other experimental settings as used in their implementation.

Point Cloud Classification We use the ModelNet40 dataset. For each instance, we uniformly sample 5000 points from each element in the dataset. We use a Set Transformer (Lee et al., 2019) with two Induced Self Attention Blocks (ISABs) in the encoder, where each ISAB contains two transformer blocks, and with a Pooling by Multihead Attention (PMA) Module in the decoder. We use an Adam optimizer, with batch size 64, 200 training epochs, and learning rate of 1×10^{-3}. For the baseline transformer model, the hidden dimensions of the ISABs are $d, k = 256$, and for the continuous-time models, they are reduced to 200. For the regularized N-ODE Transformer and OT-Transformer, the regularization hyperparameters are $\lambda = 0.1$ and $\lambda = 1$, respectively, as they provide the optimal performance in our tests. We use $T = 1$ and a total of 8 time steps for the numerical integration.

Table 6. Number of parameters for the transformer blocks, best and final test accuracies (with standard deviation) across five trials for the point cloud experiment.

Method/Exp.	**Para. Count**	**Best Test Accuracy**	**Final Test Accuracy**
Baseline	0.86M	$87.4\% \pm 0.45\%$	$86.6\% \pm 0.67\%$
Reg. N-ODE Trans.	0.65M	$87.5\% \pm 0.51\%$	$86.7\% \pm 0.43\%$
OT-Trans. (Ours)	0.65M	$\mathbf{89.9\% \pm 0.42\%}$	$\mathbf{89.3\% \pm 0.69\%}$

MNIST Classification We use a Vision Transformer (ViT) (Dosovitskiy et al., 2021) with self-attention layer with a single head. The patch size is 7×7. We use an Adam optimizer. The number of epochs is 45 and the batch size is 100. The learning rate is set to 5×10^{-4} for the first 35 epochs, then decreased to 5×10^{-5} until the 41st epoch, at which point it is reduced to 5×10^{-6}. For the baseline model, the hidden dimensions d and k are 128. For the continuous-time models, they are reduced to 64. For OT-Transformer, the regularization hyperparameter is $\lambda = 0.01$ as it provides the optimal performance in our tests. We use $T = 1$ and a total of 20 time steps for the numerical integration.

Table 7. Number of parameters for the transformer blocks, best and final test accuracies (with standard deviation) across five trials for the MNIST image classification experiment.

Method/Exp.	**Para. Count**	**Best Test Accuracy**	**Final Test Accuracy**
Baseline	93k	$93.0\% \pm 0.69\%$	$93.0\% \pm 0.67\%$
Unreg. OT-Trans.	18k	$96.8\% \pm 0.23\%$	$96.8\% \pm 0.25\%$
OT-Trans. (Ours)	18k	$\mathbf{97.1\% \pm 0.16}\%$	$\mathbf{97.1\% \pm 0.15}\%$

Cats and Dogs Classification We again use ViT. The patch size is 16×16. We use an Adam optimizer. The learning rate is 3×10^{-5}. The number of epochs is 250, and the batch size is 64. The hidden dimensions d and k are 128. For the baseline model, it has 6 transformer blocks. For the continuous-time models, the number of transformer blocks is reduced to 5. For the regularized N-ODE Transformer and OT-Transformer, the regularization hyperparameters are $\lambda = 0.005$ and $\lambda = 0.01$, respectively, as they provide the optimal performance in our tests. We use $T = 1$ and a total of 20 time steps for the numerical integration.

Table 8. Number of parameters for the transformer blocks, best and final test accuracies (with standard deviation) across three trials for the cats and dogs image classification experiment.

Method/Exp.	**Para. Count**	**Best Test Accuracy**	**Final Test Accuracy**
Baseline	1.77M	$79.3\% \pm 0.52\%$	$77.6\% \pm 0.86\%$
Unreg. N-ODE Trans.	1.48M	$76.4\% \pm 0.37\%$	$75.6\% \pm 0.48\%$
Reg. N-ODE Trans.	1.48M	$76.4\% \pm 0.30\%$	$75.6\% \pm 0.03\%$
Unreg. OT-Trans.	1.48M	$78.8\% \pm 0.63\%$	$78.2\% \pm 0.39\%$
OT-Trans. (Ours)	1.48M	$\mathbf{79.5\% \pm 0.46}\%$	$\mathbf{79.0\% \pm 0.31}\%$

Sentiment Analysis We follow (Sander et al., 2022) to use a baseline model with 6 layers of transformer blocks. For the continuous-time models, the number of transformer blocks is reduced to 3. We use an Adam optimizer with 15 epochs. The learning rate is 1×10^{-4} for the first 12 epochs and 1×10^{-5} afterward. The batch size is 64. The hidden dimensions d and k are 256. The batch size is 64. For both the regularized N-ODE Transformer and OT-Transformer, the regularization hyperparameter is $\lambda = 0.5$, as it provides the optimal performance in our tests. We use $T = 1$ and a total of 8 time steps for the numerical integration.

Table 9. Number of parameters for the transformer blocks, best and final test accuracies (with standard deviation) across five trials for for the sentiment analysis experiment. *: The unregularized continuous-time models experienced gradient explosion. And we estimate their performance by using a slight regularization $\lambda = 0.01$.

Method/Exp.	**Para. Count**	**Best Test Accuracy**	**Final Test Accuracy**
Baseline	4.74M	$83.9\% \pm 0.26\%$	$\mathbf{83.7\% \pm 0.21\%}$
Unreg. N-ODE Trans.	2.37M	$83.6\% \pm 0.40\%^*$	$83.4\% \pm 0.40\%^*$
Reg. N-ODE Trans.	2.37M	$83.9\% \pm 0.48\%$	$83.5\% \pm 0.84\%$
Unreg. OT-Trans.	2.37M	$82.7\% \pm 0.38\%^*$	$82.1\% \pm 0.89\%^*$
OT-Trans. (Ours)	2.37M	$\mathbf{84.6\% \pm 0.55\%}$	$83.7\% \pm 0.86\%$