Exploring the Frontiers of Robust and Efficient Deep Learning: Neural ODEs, Augmented Neural ODEs, and Time-Invariant Steady Neural ODEs (TISODE)

R.Srilatha *Assistant Professor, Department of Mathematics* *VNR Vignana Jyothi Institute of Engineering and Technology* *Hyderabad, Telangana* *bsrilatha82@gmail.com*	***Anitej Annabattuni*** *CSE-Artificial Intelligence and Machine Learning* *VNR Vignana Jyothi Institute of Engineering and Technology* *Hyderabad, Telangana* *a.anitej@gmail.com*	***Tanay Amboj*** *CSE-Artificial Intelligence and Machine Learning* *VNR Vignana Jyothi Institute of Engineering and Technology* *Hyderabad, Telangana* *tanayamboj@gmail.com*
Supriyo Senapati *CSE-Artificial Intelligence and Machine Learning* *VNR Vignana Jyothi Institute of Engineering and Technology* *Hyderabad, Telangana* *supriyo2k5@gmail.com*	***Vyaswanth Velchuri*** *CSE-Artificial Intelligence and Machine Learning* *VNR Vignana Jyothi Institute of Engineering and Technology* *Hyderabad, Telangana* *vvc1488@gmail.com*	***M.Sridevi*** *Assistant Professor, Department of Mathematics* *Koneru Lakshamaiah Education Foundation* *Hyderabad, Telangana* *mandaptisridevi@gmail.com*

Abstract

Neural ODEs (NODEs) are a game-changer in deep learning, going beyond the inherently discretely layered network approach by parameterizing a continuous evolution of hidden states through differential equations. This comes with several benefits, such as memory efficiency, adaptive computation, and the ability to handle irregularly sampled data [1]. However, little is known about the robustness of NODEs to input perturbations—a critical factor for real-world deployment. This paper focuses on the robustness properties of NODEs, particularly on the recently proposed Time-Invariant Steady Neural ODEs (TisODE) [2], which address these concerns. We study the theoretical underpinnings: how the time-invariant nature and embedding of the steady-state constraint within TisODE enable mitigation against both random noise and adversarial attacks. We provide empirical evaluations on several benchmark datasets and demonstrate that TisODE is a flexible drop-in module for improving robustness in a wide array of deep network architectures. We also consider the use of Augmented Neural ODEs (ANODEs) [7], given traditional NODE limitations on expressivity and efficiency. We introduce a new model, Enhanced Neural ODE Attention, which shows how continuous-time dynamics can be combined with an attention mechanism for image classification. These findings show the potential role that, more generally, NODE variants can play in the pursuit of more reliable and resilient deep learning models for real-world applications.

Keywords— ***Adversarial Attacks; Augmented Neural ODEs; Deep Learning; Neural Ordinary Differential Equations; Robustness; Time-Invariant Steady Neural ODEs***

1.INTRODUCTION

Deep learning has transformed many domains in a short span of time-from computer vision to language processing. Yet, most deep models may prove to be weak towards changes in input, and thus their use in safety-critical domains becomes a matter of concern. One of the most important parameters that, together with performance, increases the trustworthiness of deep models, is their robustness-that is, how well they handle noisy or manipulated inputs. In usual or traditional deep learning architectures, which are usually dense layered neural networks, fragility is inherent. This arises because transformations through these layers may involve complex and occasionally nonlinear computations, wherein a small input change yields a large jump in output.[1]

A very promising new direction links deep learning with dynamical systems: Neural ODEs or NODEs. Instead of the usual layer-based models, NODEs apply differential equations for hidden states and bring memory and computational advantages. They also handle irregular data much better.[1]

While the NODEs are looking promising, their robustness is still within the realm of research. In this paper, the authors do an analysis on the robustness of NODEs compared to the standard models, like CNNs. Particular attention was paid to TisODE, which possesses two major features: time invariance provides great simplification for studying state changes, while a steady-state constraint makes it more stable and less sensitive against changes within the input.

That is how this present work shows how TisODE strengthens the robustness of deep models, making NODEs an indispensable part of a reliable AI system in the future. The new model, Improved Neural ODE Attention, is introduced.[2]

Background

Deep learning has transformed many disciplines; however, traditional architectures have typically relied on ,often at great computational and memory costs Neural Ordinary Differential Equations have emerged as an interesting rescript of the above approach, parameterizing the continuous evolution of hidden states via differential equations[2], taking inspirations from residual networks.

Continuous-depth framework employs advanced numerical ODE solver to integrate dynamics of hidden state, hence yielding more flexible and thus potentially more efficient representation.

In fact, there are several concrete benefits of NODEs over their discrete counterparts. Firstly, NODEs inherently have memory efficiency since, at any instant of time in computation, they require only the most recent and next state in memory, while traditional networks store activations across all layers. Second, NODEs implicitly enable adaptive computation: given the difficulty of the dynamics, the ODE solver will naturally choose an appropriate integration step-size, sometimes resulting in considerable computational savings. NODEs naturally handle irregularly sampled data, which is probably one of the most important advantages when considering real-world applications since there are many phenomena where data collection is performed at time intervals that are not uniformly distributed[3].

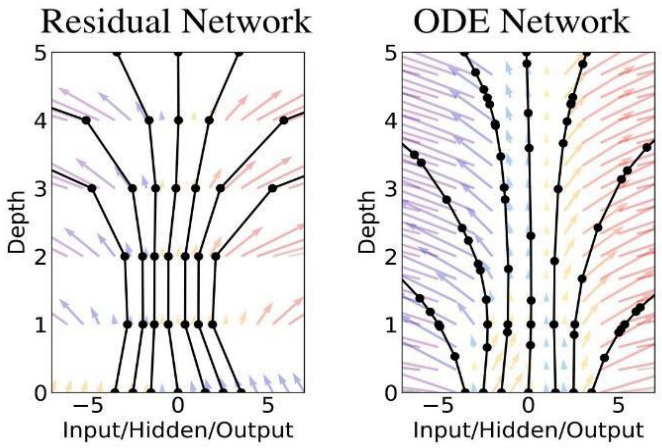

Fig.1.This figure illustrates two distinct approaches to transforming data. On the left, a Residual Network applies a series of separate, fixed transformations to the input. On the right, an ODE Network defines a continuous transformation process, akin to a vector field, that smoothly alters the data's state over time. In both illustrations, circles mark the points at which the transformations are evaluated. (Adapted from Chen et al., 2018)

2.Topological Constraints and Representational Bottlenecks of Neural ODEs

Despite some attractive computational benefits, neural ODEs inherit a variety of limitations from underlying properties of ordinary differential equations. Perhaps the most important of these constraints comes from the fact that the uniqueness of solutions of initial value problems guarantees that for an initial condition and a well-defined ODE, the system will evolve along only one, unique path. This implies that ODE trajectories naturally cannot cross in the state space. Therefore, NODEs will fail to effectively separate data points coming from different classes if these are initially intertwined or nested, since it is impossible for their trajectories to cross in order to achieve linear separability.

Limitation Of ODE Flows:

This limitation is further compounded by the homeomorphism property of ODE flows. The flow of an ODE mapping initial states to their corresponding states at a later time is a continuous function with a continuous inverse. This means that the flow is non-destructive to the topology of the input space; NODEs are confined to continuous deformations-stretching, bending, or twisting-without tearing, creating holes, or gluing parts together. These restrictions on expressivity are serious: NODEs cannot perform discontinuous transformations that might be necessary for separating complex data distributions.

These theoretical limitations have been corroborated by empirical evidence. Various works have compared NODEs with architectures not suffering from continuous deformations, such as ResNets, and convincingly showed that ResNets outperform NODEs in tasks that require changes in topology, for example, separating nested data manifolds. A simple illustrative example can be given by a 1D function; consider separating intervals [-1, 1] and [-3, -2] U [2, 3]. However, the implications are more general and concern higher-dimensional and real-world datasets.

Finally, even in those cases when a function can be approximated in principle by a NODE, the solution may require complex flows that are expensive to solve, especially for high-dimensional data. This in practice means a large number of NFEs to be performed by the ODE solver in order for the desired accuracy to be achieved. The challenge arising from another perspective of NFEs

shows that there is a need for developing methods to reduce the computational costs of NODEs without sacrificing their expressiveness**.**

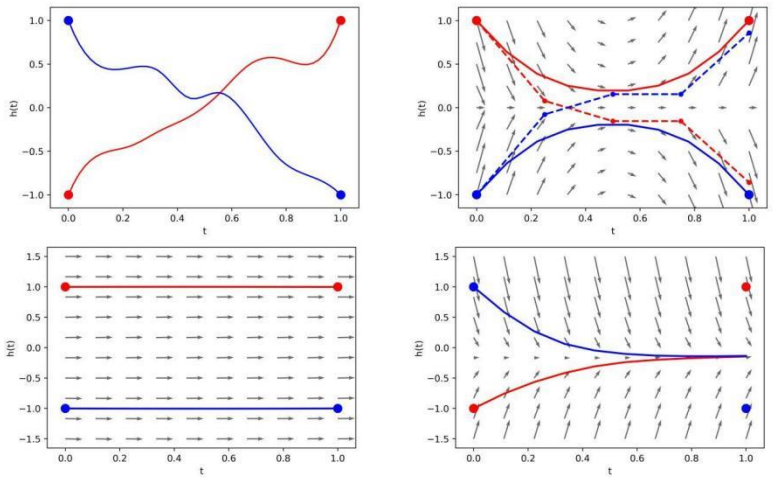

Figure 2: Continuous trajectories mapping 1 to 1 (red) and 1 to 1 (blue) must intersect each other, which is not possible for an ODE. (Top right) Solutions of the ODE are shown in solid lines and solutions using the Euler method (which corresponds to ResNets) are shown in dashed lines. As can be seen, the discretization error allows the trajectories to cross. (Bottom) Resulting vector fields and trajectories from training on the identity function (left) and g1 d(x) (right). (adapted from Dupont et al., 2019)

3.ANODE: Making Neural ODEs More Expressive and Efficient

Augmented Neural ODEs (ANODEs) address the expressiveness limitations of NODEs by augmenting the state space, increasing the model's representational capacity [7]. The core idea is to embed the input data into a higher-dimensional space, allowing the ODE to learn more expressive and simpler flows that mitigate the topological constraints of NODEs. By "lifting" the data points into additional dimensions, ANODEs can capture complex relationships and enhance performance.

Mathematical Formulation of ANODEs

Recall the standard ODE formulation in NODEs:

$$dh(t)/dt = f(h(t), t, \Theta), h(0) = x \qquad (1)$$

where:

h(t) is the d-dimensional hidden state at time t.

f(h(t))is a neural network parameterized by weights Θ , defining the system's evolution, taking the current hidden state h(t) and time t as inputs, and returning a d-dimensional vector representing the change in the hidden state over time.

x is the input data, a d-dimensional vector.

The main limitation of NODEs is the homeomorphism constraint imposed by the flow of an ODE, denoted by φt(x), mapping the initial state x to the state at time t [7]. ANODEs overcome this by augmenting the state space with an extra dimension a(t), introducing an additional ODE:

$$d/dt\ [h(t); a(t)] = f_\Theta([h(t); a(t)], t), [h(0); a(0)] = [x; 0]. \quad (2)$$

Here:

[h(t); a(t)] where h(t) is hidden state,a(t) is augmented state.

The dynamics function f takes the augmented state and time as input.The augmented dimension a(0) is initialized at zero.

The augmented flow, φ'(x) = [h(t); a(t)], is not restricted by the homeomorphism constraint due to the extra dimensions, allowing ANODEs to learn more complex functions[23].

Advantages of ANODEs

ANODEs offer several advantages over traditional NODEs:

Increased expressiveness and flexibility:

Augmented state space allows ANODEs to represent a wider range of functions, capturing more complex relationships in data [7].

Improved generalization and stability:

Simpler flows generalize better to unseen data, mitigating overfitting [7]. Moreover, they enhance training stability by reducing the complexity of learned dynamics [7].

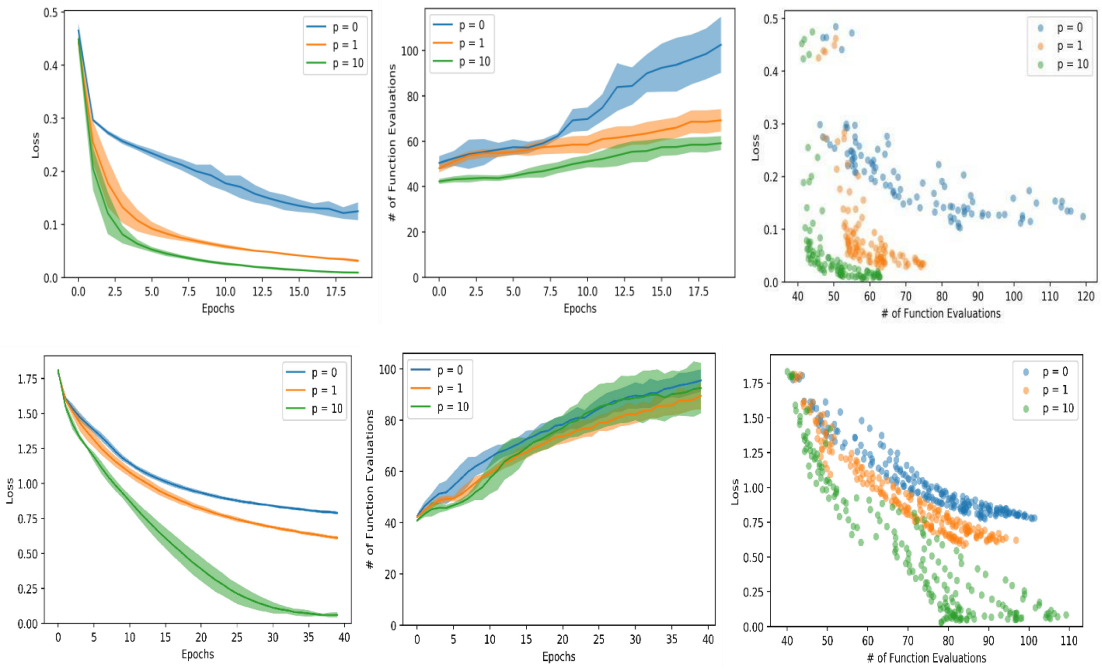

Fig.3. Training losses, NFEs and NFEs vs Loss for various augmented models on MNIST (top row) and CIFAR10 (bottom row). Note that p indicates the size of the augmented dimension, so p = 0 corresponds to a regular NODE model. Further results on SVHN and 64 × 64 ImageNet can be found in the appendix.Reference: Dupont, E., Doucet, A., & Teh, Y. W. (2019). Augmented neural ODEs. In Advances in Neural Information Processing Systems (pp. 6584-6594)

4. ROBUSTNESS ANALYSIS: CHECKING FOR RESILIENCE TO PERTURBATIONS

On neural ODEs, exhibit better robustness than conventional CNNs against random noise and adversarial attacks [2, 7]. This can be attributed to the inherent properties of ODE solutions, which tend to be resistant to small perturbations in the input data[25].

Random Noise

Random noise, often modeled as Gaussian noise, simulates real-world image degradations [2, 3]. Mathematically, we can represent the perturbation as:

$$x' = x + \varepsilon, \quad (3)$$

where:

x is the original input data, a d-dimensional vector.

x′ represents the perturbed input data, also a d-dimensional vector.

ε is a d-dimensional vector of Gaussian noise, with each element drawn independently and identically distributed from a normal distribution with mean 0 and variance σ^2[26].

Adversial Attacks

Adversarial attacks, such as the Fast Gradient Sign Method (FGSM) and Projected Gradient Descent (PGD), are designed to mislead the model by crafting carefully designed input perturbations [2, 6]. FGSM can be expressed mathematically as:

$$x' = x + \varepsilon * \text{sign}(\nabla_x L(x, y)) \quad (4)$$

where:

x is the original input data, a d-dimensional vector, on which the adversarial examples are generated.

x′ represents the adversarial example, also a d-dimensional vector.

L(x, y) denotes the loss function, measuring the difference between the model prediction and the true label y.

$\nabla_x L(x, y)$ is the gradient of the loss function with respect to input x, a d-dimensional vector pointing in the direction of the steepest ascent of the loss function.

ε is a small scalar controlling the magnitude of the perturbation.

The non-intersecting trajectory property of ODEs contributes to the inherent robustness of ODENets [26]. This stems from the uniqueness of ODE solutions for a given initial condition, ensuring bounded output deviations even with small input data perturbations. Consequently, ODE solutions are inherently resistant to noise and attacks. ODENets leverage this property, exhibiting better robustness than CNNs. ANODEs further enhance this inherent robustness due to the simpler flows they learn [2,27].

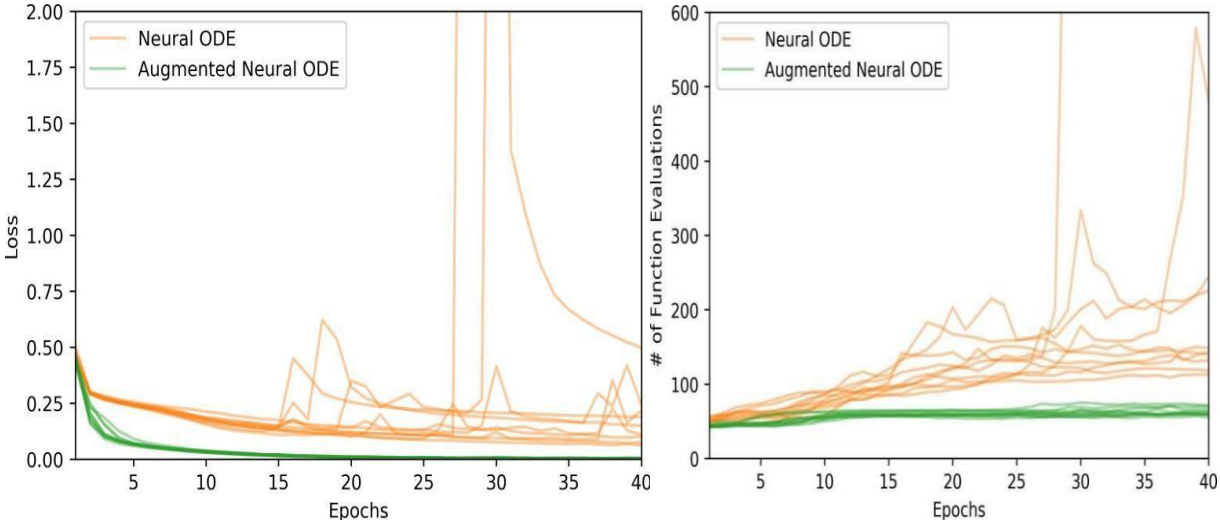

Fig. 4. Instabilities in the loss (left) and NFEs (right) when fitting NODEs to MNIST. In the latter stages of training, NODEs can become unstable and the loss and NFEs become erratic.Reference: Dupont, E., Doucet, A., & Teh. Y. W. (2019). Augmented neural ODEs. In Advances in Neural Information Processing Systems (pp. 6584-6594).

5. TISODE: BOOSTING THE ROBUSTNESS OF NEURAL ODES

5.1 Time-Invariance

TisODE removes the time dependency of the dynamics, resulting in more predictable and controllable behavior [2]. This modification can be expressed as:

$$dh(t)/dt = f(h(t), t, \Theta), h(0) = x \quad (5)$$

5.2 Constraints on Steady-State

To ensure convergence to a stable steady state, TisODE introduces a regularization term into the loss function [2]:

$$L_{ss} = \sum_1^n | \int_T^{2T} |f_\theta(Z_i(t)|dt| \quad (6)$$

where:

N is the number of training samples.

$Z_i(t)$ represents the solution to the ODE for the i^{th} sample, describing the evolution of the hidden state over time for that specific input.

T is the final integration time.

The loss term L_{ss} penalizes large state changes over extended periods, encouraging the model to settle into a stable state less sensitive to perturbations [2].

5.3 TisODE as a General Drop-in Technique: Versatility and Compatibility

Beyond its intrinsic strengths, TisODE can also act as a general "drop-in" module and be used in conjunction with

other state-of-the-art robustness methods, such as feature denoising and input randomization[2]. Interoperability further underlines its versatility and efficiency. Experiments have demonstrated that when applied on top of existing robust CNN architectures, TisODE further enhances their adversarial robustness, hence demonstrating its potential for being a general-purpose module for the enhancement of the robustness of deep networks.

6. EXPERIMENTAL RESULTS: EVALUATING THE ROBUSTNESS OF TISODE

We follow the setting of Yan et al. (2020) to test the robustness of TisODE-based classifiers under several perturbations and compare them with vanilla ODENets and CNNs.

6.1 **FGSM Adversarial Examples**

We use FGSM as a gradient-based attack method for generating the adversarial examples [2, 4].

6.1.1 **Random Gaussian Noise**

We add Gaussian Noise to the input data for modeling random noise degradation.[2,3]

6.1.2 **TisODE: A General-purpose Plugin**

Beyond standalone capabilities, TisODE can also function as a "drop-in" module that easily interoperates with other cutting-edge robustness techniques, such as feature denoising and randomizing the input [2, 10]. This modularity further enhances its versatility and effectiveness.

Table I: Summary of the classification accuracy for the TisODE-based models in comparison to vanilla ODENets and CNNs across MNIST, SVHN, and ImgNet10 datasets. The results have proved the improvement brought by TisODE towards

Dataset	Perturbation	CNN	ODENet	TisODE
MNIST	σ = 100	98.7±0.1	99.4±0.1	99.6±0.0
	FGSM-0.3	54.2±1.1	71.5±1.1	75.7±1.4
	FGSM-0.5	15.8±1.3	19.9±1.2	26.5±3.8
	PGD-0.2	32.9±3.7	64.7±1.8	67.4±1.5
	PGD-0.3	0.0±0.0	13.0±0.2	13.2±1.0
SVHN	σ = 35	90.6±0.2	95.1±0.1	94.9±0.1
	FGSM-5/255	25.3±0.6	49.4±1.0	51.6±1.2
	FGSM-8/255	12.3±0.7	34.7±0.5	38.2±1.9
	PGD-3/255	32.4±0.4	50.9±1.3	52.0±0.9
	PGD-5/255	14.0±0.5	27.2±1.4	28.2±0.3
ImgNet10	σ = 25	92.6±0.6	92.6±0.5	92.8±0.4
	FGSM-5/255	40.9±1.8	42.0±0.4	44.3±0.7
	FGSM-8/255	26.7±1.7	29.0±1.0	31.4±1.1
	PGD-3/255	28.6±1.5	29.8±0.4	31.1±1.2
	PGD-5/255	11.2±1.2	12.3±0.6	14.5±1.1

Key Observations of Table-1:

MNIST: It is always ahead of vanilla ODENets,and gains made against FGSM and PGD attacks are very obvious

SVHN: TisODE obtains better accuracies against all type of adversial examples

ImgNet10: Although less prominent overall,one can see an advantage of TisODE against stronger attacks by PGD..

7. ENHANCED NEURAL ODE ATTENTION: A PARAMETER EFFICIENCY APPROACH FOR IMAGE CLASSIFICATION

Our Enhanced Neural ODE Attention proposes a new architecture that integrates continuous-time modeling through Neural ODEs with attention mechanisms for efficiently learning complex image temporal patterns in parameters.

7.1 Motivation and Design Philosophy

This model addresses the limitations of classic discrete-layered attention models by incorporating:

7.1.1 ODE Dynamics:

We use, instead, a system of ODEs to model the continuous evolution of hidden representations for capturing richer, finer-grained temporal patterns, which help in the extraction of more informative feature representations.

7.1.2 Multi-Timestep Attention for Enhanced Temporal Context:

The model applies attention mechanisms at

multiple time steps along the ODE trajectory, enabling the aggregation of contextual information at different points in the temporal evolution of hidden states, providing a holistic view of the input data.

7.1.3 Efficient Training:

We utilize mixed-precision training with FP32 and FP16 computations to reduce memory usage and accelerate training, particularly beneficial for large models [14]. Layer normalization within the ODE function and dropout after context aggregation further enhance training stability and prevent overfitting.

7.2 Architecture and Implementation

The core components of Enhanced Neural ODE Attention are:

7.2.1 Input Encoding:

The input image is pre-processed and transformed into a sequence-like format, either by flattening or dividing it into patches. A fully connected layer (nn.Linear) then maps the sequential input to a hidden dimension, encoding the input in a feature space suitable for ODE processing.

7.2.2 ODEFunc (System of Ordinary Differential Equations):

This class defines the system of ODEs governing the continuous evolution of latent states. The structure of the parameterized ODE system is defined using standard neural network building blocks like linear layers (nn.Linear), layer normalization (nn.LayerNorm), and rectified linear unit activations (nn.ReLU).

7.2.3 Odeint:

We use the odeint solver from the torchdiffeq library to solve the defined ODE system [1]. This solver takes the initial hidden state and integrates the ODE system over a specified time interval, returning a sequence of hidden states representing the trajectory of latent state evolution in time.

7.2.4 Multi-Timestep Attention:

Hidden states are extracted at specific time points, with independent attention mechanisms applied to focus on different aspects of the input at each timestep.

7.2.5 Attention Mathematical Formulation:

Let h_t be the hidden state at time step t, and W_q, W_k, and W_v be learnable weight matrices. The attention mechanism can be formulated as:

Query: $Q = h_t W_q$ (7)

Key: $K = h_t W_k$ (8)

Value: $V = h_t W_v$ (9)

Attention Weights: $A = \text{softmax}(QK^T/\sqrt{d_k}).V$, where d_k is the key dimension. (10)

Context Vector: $C_t = AV$ (11)

7.2.6 Image Classification Output:

The context vectors from each time step are combined into a single context representation, often by averaging. A linear decoder (nn.Linear) is then applied to this aggregated context representation to generate a probability distribution over image classes, enabling image classification.

7.3 Implementation Details and Performance Optimization

To enhance efficiency and stability, we implement:

7.3.1 Mixed-precision training:

We utilize torch.cuda.amp.autocast and torch.cuda.amp.GradScaler to perform mixed-precision training [14].

7.3.2 Layer normalization:

We apply layer normalization (nn.LayerNorm) within the ODEFunc class to stabilize activations and mitigate vanishing or exploding gradients.

7.3.3 Dropout:

Dropout is applied after the context aggregation to avoid overfitting and to regularize the model. Classic Attention tends to narrowly focus on one position of the input sequence across datasets

like MNIST, FashionMNIST, CIFAR10, and CIFAR100 setting high attention for one feature and ignoring the contextual information from other places.

That would mean that Classic Attention, due to architectural biases or poor choices of hyperparameters which only minimize its capability of capturing global dependencies, has been reduced to a feature selector essentially.On the contrary, Neural ODE Attention reflects much more dynamic and adaptive behavior: while attending to multiple positions in a sequence, it manages to capture complex temporal dependencies. It can merge information much better this way, providing an understanding of the input data in more detail. First, notice how different the patterns of attention are for various datasets, such as MNIST and CIFAR10, which is indicative that Neural ODE Attention does adapt to specific features of a dataset. While these kinds of visualizations make sense intuitively, proper quantitative metrics are to be done in order to verify these results. Still, early results would indicate that Neural ODE Attention is far more expressive and adaptive than Classic Attention.

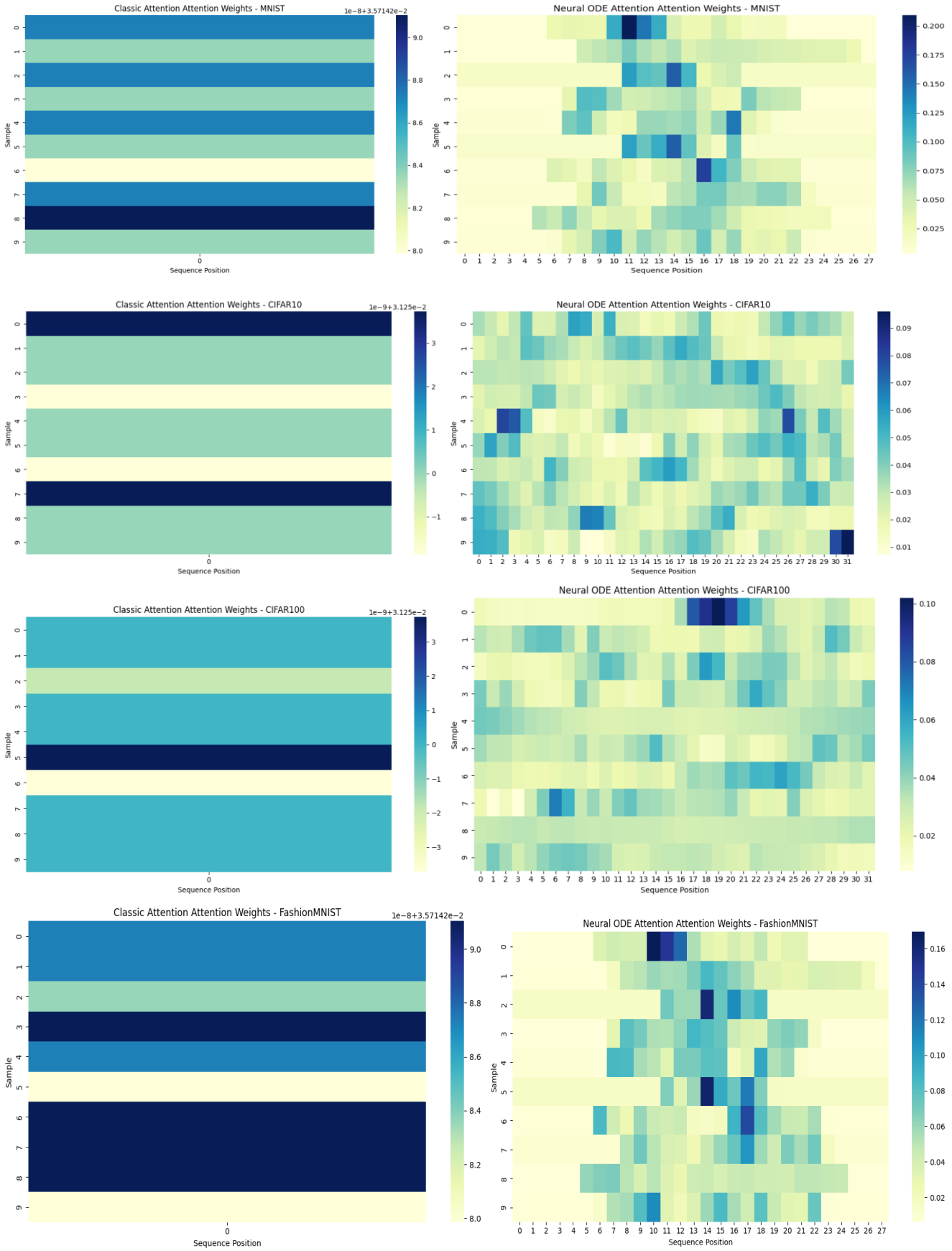

Fig.5. Representation of attention plots of both the models trained on various datasets.

7.4 Experimental Setup and Results

We evaluate the performance of Enhanced Neural ODE Attention on four benchmark image classification datasets: MNIST, FashionMNIST, CIFAR-10, and CIFAR-100. We compare the model against a similarly optimized Classic Attention model (without the ODE component) as a baseline.

Table III: Performance Results on Image Datasets based on inference time

Model	Dataset	Test Loss	Accuracy	Precision	Recall	F1 Score	Inference time(s)
Neural ODE Attention	**MNIST**	**0.3699**	**0.8874**	**0.8869**	**0.8874**	**0.8868**	**0.0955**
	FashionMNIST	**0.3800**	**0.8874**	**0.8863**	**0.8874**	**0.8866**	**0.1031**
	CIFAR10	**1.7411**	**0.3822**	**0.3730**	**0.3822**	**0.3737**	**0.1038**
	CIFAR100	**3.9617**	**0.1233**	**0.1032**	**0.1233**	**0.0991**	**0.1069**
Classic Attention	**MNIST**	**0.3863**	**0.8828**	**0.8827**	**0.8828**	**0.8826**	**0.9990**
	FashionMNIST	**0.385128**	**0.8842**	**0.884171**	**0.8842**	**0.8840**	**1.0212**
	CIFAR10	**1.775356**	**0.3603**	**0.352789**	**0.3603**	**0.3533**	**1.0038**
	CIFAR100	**4.053468**	**0.1025**	**0.091688**	**0.1025**	**0.077547**	**1.013438**

Table IV. Performance Based on Parameters

Dataset	Classic Attention Model	Neural ODE Attention Model
MNIST	**5387**	**3531**
Fashion MNIST	**5387**	**3531**
CIFAR10	**5899**	**3659**
CIFAR100	**17509**	**6629**

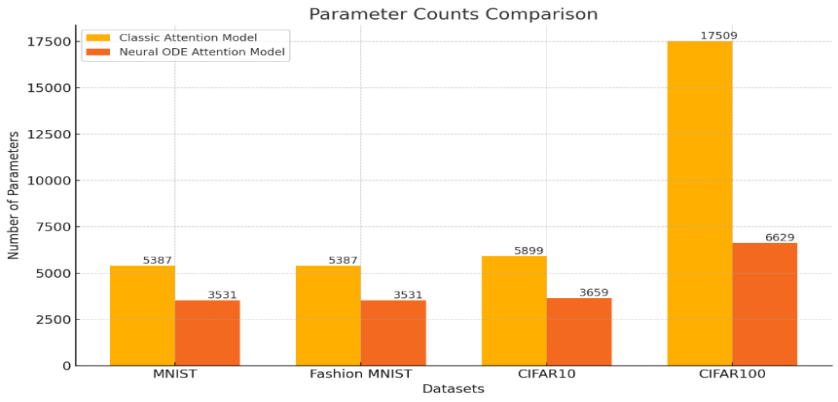

Figure.6. Parameter Comparison

Exploring different ODE solvers and integration schemes:

The code for enhanced Neural ODE model is publicly available here https://www.kaggle.com/code/aniteja/neural-ode-attention. Researchers can use it to explore depth learning.

Tables 3 and 4 show the results, demonstrating the promise of Enhanced Neural ODE Attention for image classification. The model achieves competitive accuracy compared to Classic Attention on simpler datasets while using significantly fewer parameters, highlighting its parameter efficiency.

7.5 *Discussion and Future Directions*

The results highlight the potential of Neural ODEs for data with strong temporal dependencies. Future research directions include Investigating alternative solvers and schemes can significantly impact model performance and computational efficiency.

7.5.1 Adaptive time stepping:

Adapting the integration step size based on the complexity of the dynamics can improve accuracy and efficiency

7.5.2 Hybrid architectures:

Combining Neural ODEs with other architectural components, such as convolutional layers, can further enhance expressiveness for image classification tasks [13].

7.5.3 Generalizing to other application domains:

The model can be applied to other sequential data tasks in natural language processing, time series analysis, and video understanding.

8. DISCUSSION AND FUTURE WORK: PUSHING THE FRONTIERS OF CONTINUOUS-DEPTH LEARNING

8.1 Open Challenges and Limitations

8.1.1 Computational Cost:

Evaluating NODEs remains computationally intensive[22], even with ANODEs. More efficient solvers and alternative training strategies are needed for scalability..

8.1.2 Hyperparameter Tuning:

ANODEs and TisODE introduce new hyperparameters requiring careful tuning for optimal performance. Principled approaches for hyperparameter selection need to be developed.

8.1.3 Possible Future Research Directions

8.1.3.1 Exploring alternative strategies to overcome NODE representational limitations:

It is also important to investigate methods beyond augmentation that could mitigate expressiveness limitations in NODEs, such as learning-based augmentation, stochasticity in dynamics[22,24], or generalization of augmentation techniques to other deep learning models

8.1.4 Developing a theoretical framework for robustness analysis:

A rigorous theoretical framework for the analysis of NODE robustness would give more insight into their behavior and guide architectural choices

8.1.5 Investigating robustness-accuracy trade-offs:

Understanding and controlling the trade-off between robustness and accuracy in NODEs is important. Development of metrics and methods to measure and estimate this tradeoff would facilitate making informed design decisions [16].

8.1.6 Improving both robustness and accuracy:

One of the most promising ways is to explore training strategies and architectural modifications that can improve robustness and accuracy simultaneously.

9.References

[1] Chen, R. T. Q., Rubanova, Y., Bettencourt, J., & Duvenaud, D. K. (2018). Neural ordinary differential equations. Advances in Neural Information Processing Systems, 31, 6571–6583.

[2] Yan, H., Du, J., Tan, V. Y. F., & Feng, J. (2020). On robustness of neural ordinary differential equations. International Conference on Learning Representations.

[3] Szegedy, C., Zaremba, W., Sutskever, I., Bruna, J., Erhan, D., Goodfellow, I. J., & Fergus, R. (2013). Intriguing properties of neural networks. arXiv preprint arXiv:1312.6199.

[4] Goodfellow, I. J., Shlens, J., & Szegedy, C. (2014). Explaining and harnessing adversarial examples. arXiv preprint arXiv:1412.6572.

[5] He, K., Zhang, X., Ren, S., & Sun, J. (2016). Deep residual learning for image recognition. Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 770–778.

[6] Madry, A., Makelov, A., Schmidt, L., Tsipras, D., & Vladu, A. (2017). Towards deep learning models resistant to adversarial attacks. arXiv preprint arXiv:1706.06083.

[7] Dupont, E., Doucet, A., & Teh, Y. W. (2019). Augmented neural odes. Advances in Neural Information Processing Systems, 32, 3134–3144.

[8] Coddington, E. A., & Levinson, N. (1955). Theory of ordinary differential equations. Tata McGraw-Hill Education.

[9] Younes, L. (2010). Shapes and diffeomorphisms (Vol. 171). Springer Science & Business Media.

[10] Xie, C., Wu, Y., van der Maaten, L., Yuille, A. L., & He, K. (2019). Feature denoising for improving adversarial robustness. Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 501–509.

[11] Xie, C., Wang, J., Zhang, Z., Ren, Z., & Yuille, A. (2017). Mitigating adversarial effects through randomization. arXiv preprint arXiv:1711.01991.

[12] Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., ... & Polosukhin, I. (2017). Attention is all you need. Advances in neural information processing systems, 30, 5998-6008.

[13] Lu, J., Zhong, A., Li, J., & Dong, J. (2019). Deep pruning of neural networks with adaptive batch normalization. arXiv preprint arXiv:1908.03930.

[14] Kingma, D. P., & Ba, J. (2014). Adam: A method for stochastic optimization. arXiv preprint arXiv:1412.6980.

[15] Grathwohl, W., Chen, R. T. Q., Bettencourt, J., Sutskever, I., & Duvenaud, D. (2018). Ffjord: Free-form continuous dynamics for scalable reversible generative models. arXiv preprint arXiv:1810.01367.

[16] Massaroli, S., Poli, M., Bin, M., Park, J., Yamashita, A., & Asama, H. (2020). Stable neural flows. arXiv preprint arXiv:2003.08063.

[17] Massaroli, S., Poli, M., Park, J., Yamashita, A., & Asama, H. (2020). Dissecting neural odes. arXiv preprint arXiv:2002.08071.

[18] Finlay, C., Jacobsen, J.-H., Nurbekyan, L., & Oberman, A. M. (2020). How to train your neural ODE. arXiv preprint arXiv:2002.02798.

[19] Greydanus, S., Dzamba, M., & Yosinski, J. (2019). Hamiltonian neural networks. Advances in Neural Information Processing Systems, 32, 15379–15389.

[20] Lutter, M., Ritter, C., & Peters, J. (2019). Deep lagrangian networks: Using physics as model prior for deep learning. arXiv preprint arXiv:1907.04490.

[21] Poli, M., Massaroli, S., Park, J., Yamashita, A., Asama, H., & Park, J. (2019). Graph neural ordinary differential equations. arXiv preprint arXiv:1911.07532.

[22] Poli, M., Massaroli, S., Yamashita, A., Asama, H., & Park, J. (2020). Hypersolvers: Toward fast continuous-depth models. arXiv preprint arXiv:2007.09601.

[23] Toth, P., Rezende, D. J., Jaegle, A., Racanière, S., Botev, A., & Higgins, I. (2019). Hamiltonian generative networks. arXiv preprint arXiv:1909.13789.

[24] Liu, X., Si, S., Cao, Q., Kumar, S., & Hsieh, C.-J. (2019). Neural SDE: Stabilizing neural ODE networks with stochastic noise. arXiv preprint arXiv:1906.02355.

[25] Carlini, N., & Wagner, D. (2017, May). Towards evaluating the robustness of neural networks. In 2017 ieee symposium on security and privacy (sp) (pp. 39-57). IEEE.

[26] Athalye, A., Carlini, N., & Wagner, D. (2018). Obfuscated gradients give a false sense of security: Circumventing defenses to adversarial examples. arXiv preprint arXiv:1802.00420.