Xingyu Zhu[1] and Xiaoming Shi[1]

[1]Division of Environment and Sustainability, Hong Kong University of Science and Technology

December 27, 2023

Abstract

Data-driven parameterization schemes have gained much attention in the field of atmospheric science nowadays. We design an ideal case using the Barotropic Vorticity Equation (BVE) model with periodic shear forcing and a deep learning (DL) model. The trained BVE model can greatly improve its initial forecast lead time with the data-driven parameterization scheme. However, the challenge is the significant amount of time required to employ the model, which is a hurdle for practical applications. Thus, this research aims at enhancing the model's efficiency while minimizing any negative impact on its performance. Here we propose three methods, compressing the amount of model parameters (CNNhk), compressing the amount of training data (SGM), and combining both of them (CFS). Through these three methods and utilizing the solver-in-the-loop (SOL) method, we successfully reduce the model's runtime while preserving its initial performance. By improving the effectiveness of our model, we believe it can contribute to the development of more efficient data-driven parameterization schemes and inspire further explorations.

Data-Driven Parameterization for A Subgrid-Scale Model: Methods for Improving Model Efficiency

Xingyu Zhu [1]**and Xiaoming Shi** [1*]

[1]Division of Environment and Sustainability, Hong Kong University of Science and Technology, Hong Kong.

Key Points:

- Data-driven parameterization schemes using deep learning models are proved useful but sometimes are still not efficient enough.
- We set up the case with a periodically perturbed model using barotropic vorticity equation and a convolutional neural network as the parameterization scheme.
- Three new methods are proposed to refine the original CNN model for improving its runtime while keeping its initial model capacity.

*shixm@ust.hk

Corresponding author: Xiaoming Shi, `shixm@ust.hk`

Abstract

Data-driven parameterization schemes have gained much attention in the field of atmospheric science nowadays. We design an ideal case using the Barotropic Vorticity Equation (BVE) model with periodic shear forcing and a deep learning (DL) model. The trained BVE model can greatly improve its initial forecast lead time with the data-driven parameterization scheme. However, the challenge is the significant amount of time required to employ the model, which is a hurdle for practical applications. Thus, this research aims at enhancing the model's efficiency while minimizing any negative impact on its performance. Here we propose three methods, compressing the amount of model parameters (CNNhk), compressing the amount of training data (SGM), and combining both of them (CFS). Through these three methods and utilizing the solver-in-the-loop (SOL) method, we successfully reduce the model's runtime while preserving its initial performance. By improving the effectiveness of our model, we believe it can contribute to the development of more efficient data-driven parameterization schemes and inspire further explorations.

Plain Language Summary

There are many atmospheric motions with such tiny space scales that they cannot be resolved by the unified resolution used in numerical models, which are called subgrid-scale (SGS) processes. SGS processes, such as turbulence, is important for improving weather forecast ability. Instead of using traditional parameterization schemes, nowadays data-driven schemes are explored. However, the problem is the huge computational time needed for running complex DL models. So in this work, we first use the BVE model with periodic SGS process generated to build up an ideal case, and design a data-driven parameterization scheme with the initial DL model. Then we design three new methods for adjustments of the initial DL model to reduce the computational time consumption. Moreover, while speeding up DL models' runtime, we maintain the initial model capacity by including more physical information of BVE to the trained DL models. Through this work, we hope to contribute to and motivate the explorations for data-driven parameterization schemes.

1 Introduction

Appropriately handling with the calculation of sub-grid scale (SGS) processes are critical to enhance the ability of numerical weather forecast models. There are various complex physical processes with significantly different space scales to be calculated in a weather forecast model, thus some tiny processes cannot be resolved by the unified resolution, which are also called SGS processes. Despite their minuscule size and rapid evolution, SGS processes exert a substantial influence on weather forecast accuracy, weather patterns and even climate dynamics (Müller & Scherer, 2005; Alapaty et al., 2012; Sherwood et al., 2014). Traditionally, parameterization schemes are applied to solve this problem in numerical weather predictions (NWP). However, since SGS processes are usually nonlinear and hard to be described by simple formulas, these traditional schemes which are based on assumptions can add great uncertainties to the stability of numerical models and cause unpredictable biases in calculations (Goger et al., 2016; Mantovani Júnior et al., 2023).

Data-driven models can serve as potent instruments in such circumstances. Firstly, they run extraordinarily fast. Benefiting from the rapid advancement of computer hardware and software, data-driven models, employing diverse neural networks (NN), possess the capability to efficiently undergo training using vast datasets within shorter timeframes when compared to conventional numerical models. PanGu (Bi et al., 2023), GraphCast (Lam et al., 2022) and FengWu (Chen et al., 2023) are outstanding representatives of data-driven models for medium-range global weather forecasting, showing promising prospects for the utilization of deep learning in the field of atmospheric sciences. What's more, data-driven models are somehow able to learn physical patterns from partial differential equations (PDEs). Previous work includes physics-informed neural networks (PINN) (Raissi et al., 2019), solver-in-the-

loop (SOL) method (Um et al., 2020), and learning neural operators (NO), such as DeepONet (Lu et al., 2021) and FNO (Kovachki et al., 2021). Among them, the SOL method allows DL models to interact with physical constraints by integrate physical information into the training process, which implies that these models can rely not solely on data itself but also incorporate the fundamental principles of physics. Therefore, parameterization schemes trained by data-driven models tend to get faster but still accurate simulation results, as evidenced by the findings of Kochkov et al. (2021), where the numerical calculations for the Kolmogorov flow are augmented with a DL model. Following the former work, Qu and Shi (2023) also get the same results by using a convolutional neural network (CNN) to train an idealized 2-D case with the barotropic vorticity equation (BVE), which is more similar to the real atmosphere mode compared to the Kolmogorov flow.

However, though existed work using data-driven models has obtained obvious improvements of extended forecast lead time, the time consumption for the trained model to be applied for predictions is still not so satisfactory due to the huge number of parameters from deep learning neural networks. This problem presents a major obstacle to the practical application of data-driven parameterization schemes in real numerical weather forecast models in the future, because parameterization schemes should play a small role in the overall weather forecast model and should not significantly increase computational time. As a result, how to realize effective compression of runtime consumption is a meaningful technical problem to be handled.

Our work will focus on the solutions of enhancing the model's efficiency while minimizing any negative impact on its performance. We first set up the same ideal case from Qu and Shi (2023) and downscale the initial data generated from the simulation to build up the dataset with a long period. Then we test several new losses which can significantly extend forecast lead time of our BVE model, one of which is used as a baseline for later comparisons. At the final part, we propose three different methods to achieve our goal in this article. We conclude our three methods to be classified as newly designing a more compact model, using less training data and both utilizing a compact model as well as informative training data.

2 Experiment Design and Methods

2.1 Case Introduction and Data

Same as the case designed by Qu and Shi (2023), we use the barotropic vorticity equation to describe the vorticity evolution of a two-dimensional flow, and activate periodic instability by adding a shear forcing to the middle domain of the flow at regular intervals, which is around every 10 time units. Figure 1a visualizes the vorticity evolution pattern and demonstrates how the forecast errors occur due to SGS processes in our case. The domain is characterized by large eddies for the majority of the time, but once the periodic shear forcing is added, SGS processes like turbulence, which cannot be resolved by a large grid size, are generated. Thus the LowRes simulation without a parameterization scheme to solve the SGS processes performs obvious forecast biases within short running steps, but the CNN8 simulation with a data-driven parameterization scheme can still get a satisfactory prediction result after the same integral time.

Also, instead of relying on a traditional deep-learning model which directly describes its loss function with the image-to-image error, we continuously utilize the SOL method proposed by Um et al. (2020) with a JAX framework (Bradbury et al., 2018) to integrate the PDE solver with deep learning neural networks and do automatic differentiation. This is due to two main factors. Firstly, we observe subpar online test forecast results without the employment of the SOL method, which demonstrate even poorer performance than the baseline (LowRes) though there appears to be normal gradient descent for the loss function during the training process. Moreover, the SOL method has another benefit that it allows extended forecast lead time for the same model by using longer look-ahead-steps in training loops (Figure

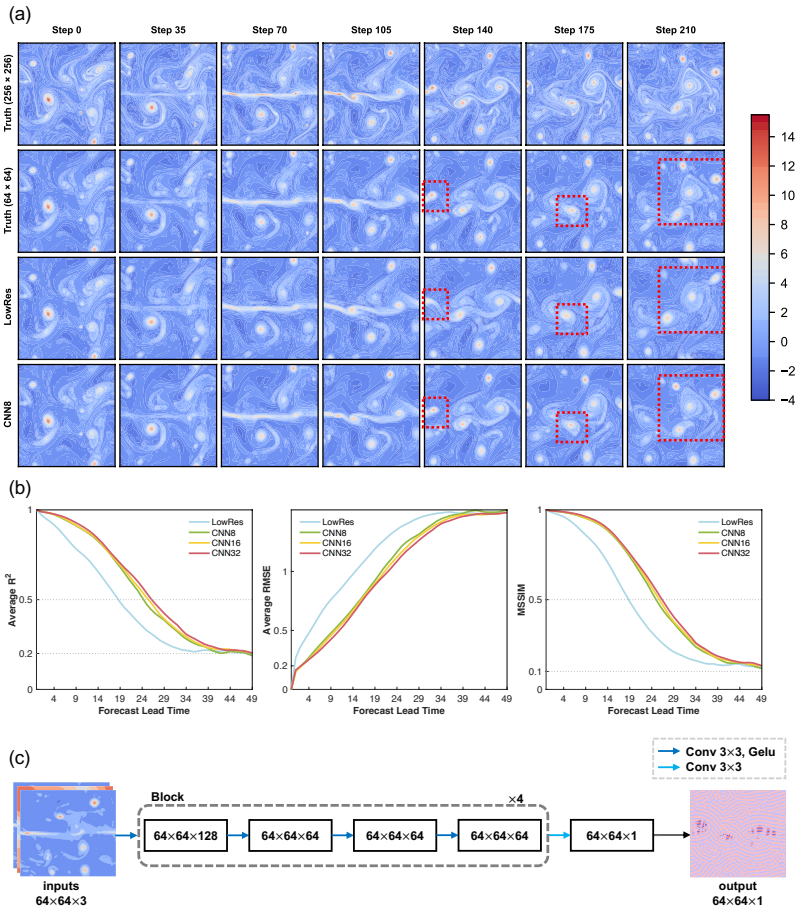

Figure 1. Snapshots of vorticity for the truth and the predictions running with a low resolution (a), the averaged R^2, RMSE and MSSIM curves for the initial CNN model with different look-ahead time steps, where R is the correlation coefficient, MSSIM is the multi-scale structural similarity, and the number following the name of the model in the legend represents the number of the applied look-ahead-steps (b), and the architecture of the initial CNN model (c). For (a), the first two rows are the truth run, separately with an initial resolution (Truth (256×256)) and a coarsened one (Truth (64×64)), and the last two rows are the predictions running on the low resolution of 64×64 with the same initial condition from Truth (64×64), separately one without any parameterization scheme (LowRes) and one with a data-driven parameterization scheme using the initial CNN model with 8 look-ahead steps (CNN8).

1b), which provides a pragmatic solution for the seemingly unavoidable accuracy loss when compressing the amount of model parameters or training data for less time consumption. As a result, 8 look-ahead-steps will be used for the baseline, and the initial CNN model proposed in the referenced paper (Figure 1c) will be the foundational architecture for modifications. All the refined models will be trained with longer look-ahead-steps, for which 32 is chosen in this work, and compared to the baseline both in the respect of forecast lead time and calculation time consumption.

For the dataset, a long period of truth run with an initial resolution of 256×256 is generated every time step with $\Delta t = 0.01$. Then it is coarsen-grained into a resolution of 64×64 every 5 time steps, and $\Delta t = 0.05$ will be used for every step of integration for all simulations on

this coarse grid, which is to say every time unit includes 20 time integrals of simulations running on the low resolution. This coarsened truth is used as the overall dataset for later experiments, including pretraining and online testing. The first 50,000 time steps of the overall dataset, equal to 2500 time units or 250 forcing cycles, are chosen for pretraining, 80% of which for training and 20% for validation. Then 54,000 $\sim$ 64,000 time steps of the overall dataset are used for online testing. Specifically, the online testing is carried out for 5 samples every 100 time units and evaluated by the average of them to comprehensively consider the model forecast ability for different time sets. Further considering forecast ability biases among different initial vorticity patterns, ensembles are also generated. For each sample, one every other time unit is selected as the initial condition for simulations. Thus 50 ensembles are contained in one sample and each sample is calculated with the average status of the 50 ensembles' prediction results. More details about the former work, such as the derivation for the governing equation of BVE model and the original design of the CNN model, can be referred to from Qu and Shi (2023).

2.2 Loss Design

Except for the design of case and dataset, the optimization problem also plays a key role in deep learning, especially when it is employed for physics-related tasks. Thus firstly some useful modifications are made to the original loss function. Instead of directly using the original loss:

$$\mathcal{L}_{original} = \frac{1}{N}\sum_{k}^{N} \boldsymbol{L}(\mathcal{M}^k \boldsymbol{x}_{t_0}, \boldsymbol{x}_{t_k}), \tag{1}$$

it is improved to be a scaled one:

$$\begin{aligned} \beta &= \log_{10} \boldsymbol{L}(\mathcal{M}^k \zeta_{t_0}, \zeta_{t_k}) - \log_{10} \boldsymbol{L}(\mathcal{M}^k \psi_{t_0}, \psi_{t_k}) \\ \mathcal{L}_{scaled} &= \frac{1}{N}\sum_{k}^{N} (\boldsymbol{L}_{\zeta_k} + 10^{\beta} \boldsymbol{L}_{\psi_k}), \end{aligned} \tag{2}$$

where N is the number of look-ahead-steps used for training and $\boldsymbol{x}$ is the tensor of the general physical state consisting of ζ (vorticity) and ψ (stream function). $\mathcal{M}$ represents the DL model and $\boldsymbol{L}$ is the mean square error (MSE) calculated in each look-ahead-step. $\mathcal{L}$ is the final loss function which is the averaged accumulation of $\boldsymbol{L}$. The design for the scaled loss originates from the thought similar with that of PINNs. For PINNs, various differential equation residuals with different weights are considered into the loss function to include more physical information into the training process (Cuomo et al., 2022). Here a scaling parameter β is applied to split the original $\boldsymbol{L}$ term, which generally represents the physical state, and strengthen the impact of the stream function for the training trajectory because significant difference in order of magnitude for $\boldsymbol{L}_{\zeta}$ and $\boldsymbol{L}_{\psi}$ is observed, which can be around $10^4 \sim 10^6$.

Another interesting thing we find is that though periodic boundary conditions are applied for paddings, the pretrained model still exhibits deficiencies in giving concise predictions at boundaries of the domain, both in terms of locations and intensity. Thus another newly designed loss is further tested based on the scaled loss:

$$\begin{aligned} f(y) &= \cos(\frac{4\pi}{63} y) + 2 \quad (0 \le y \le 63) \\ \mathcal{L}_{cos} &= f(y) \times \mathcal{L}_{scaled} \end{aligned} \tag{3}$$

where y is the count of rows in y direction, and the cos loss is calculated by adding a cos-shaped amplification function to the scaled loss in y direction, which aims at driving DL models to give more attention to the evolution of physical states at boundaries.

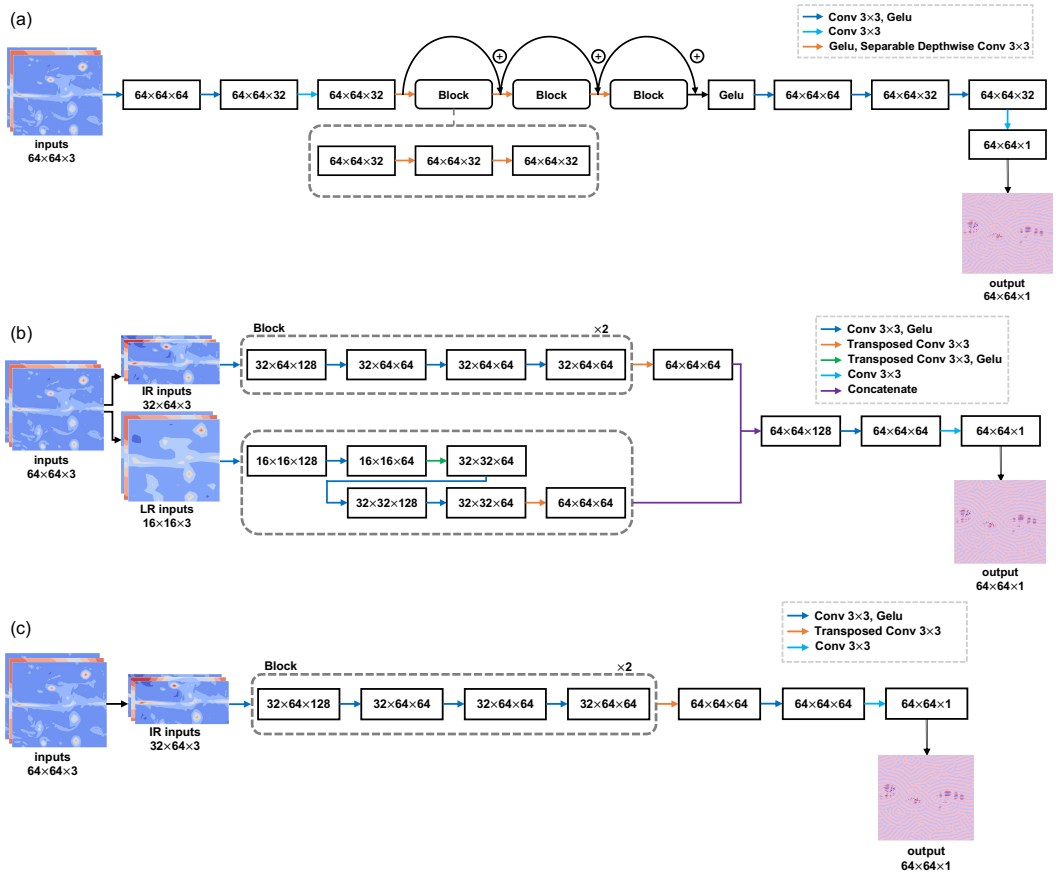

Figure 2. Model structures for CNNhk (a), SGM (b), and CFS (c), where IR inputs are inputs with the initial resolution, and LR inputs are inputs with a coarser resolution.

2.3 Methods for Model Efficiency Improvement

To enable less computational time for application, the basic idea is considered as either to compress the model or to compress the data for computing. Based on this principal, three kinds of methods are proposed and can be classified as using a more compact model, using a model with smaller training data, and both using a more compact model with smaller training data.

The newly-designed model which is more compact than the initial one is named CNNhk (Figure 2a). Separable depthwise convolution with residual blocks are used in this model. Such model structures have been employed for many tasks, such as Xception (Chollet, 2017), a refined ResNet-26 (Guo et al., 2019) and R**x**Net (Shi, 2020). It has been proved in former experiments that such model structures can effectively make a compact model without obvious losses in the model capacity. Thus by using separable depthwise convolution with residual blocks and fewer convolutional layers, the number of model parameters in CNNhk is compressed and it is expected that the computational time can be evidently reduced without much loss in the model forecast ability.

The second method which is named a split-grid method (SGM, Figure 2b) emulates from the architecture of the weather research and forecast (WRF) model, where multiple nested domains with different resolutions and extents are applied. Similarly in SGM, the original input domain is split into two parts, one only containing the central part of the original do-

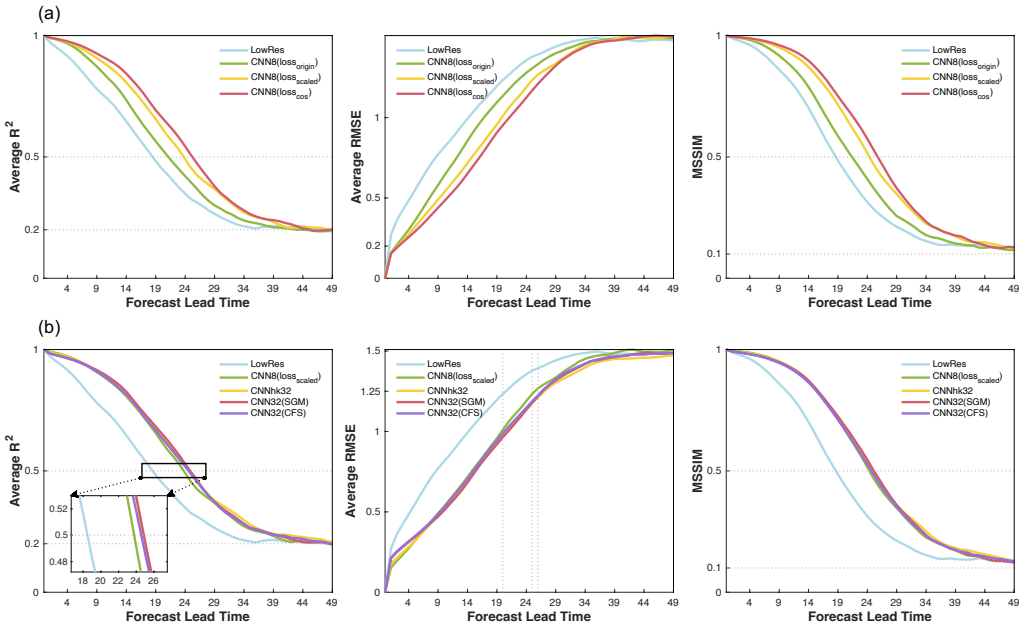

Figure 3. Curves of averaged R^2, RMSE and MSSIM over time for the initial CNN model with different loss functions (a) and for the refined models (b), where R is the correlation coefficient and MSSIM is the multi-scale structural similarity.

main where periodic shear forcing is most prominently activated with the initial resolution, and the other covering the whole domain but with a coarser resolution. This designed structure allows less data used for training process and thus can effectively reduce the number of model parameters. But meanwhile, keeping the resolution for the key area where SGS processes are most generated can refine the model to retain the most important information, thereby ensuring the sustained excellence of forecast quality, which indicates the DL model with SGM can mitigate excessive information loss when the amount of input data is reduced.

Inspired by SGM, the third method named a central-focus scheme (CFS, Figure 2c) is further considered. This scheme only trains the central part of the initial domain and aims to explore if it is practicable to solely train the physical states in the key area to compress the input data as well as the number of DL model layers.

3 Results

Figure 3a shows the effectiveness of the newly designed loss functions. The averaged R^2 with its value equal to 0.5 is set as the threshold, beyond which the predictions are regarded as still robust. MSSIM results, which provide evaluation metrics for structural similarities (Z. Wang et al., 2003), are also shown to corroborate with R^2. It is evident that using the same DL model, the scaled loss performs much better than the original one, and the cos loss can extend extra valid forecast lead time of the scaled loss, with improvement around 10%, 25% and 30% compared to LowRes separately. Additionally, another loss function that only calculates the accumulated errors of the stream function is also tested (not shown) and can have a similar level of performance with the scaled one for the initial DL model. However, when it is multiplied with the cos-shaped function or applied to the refined models, the model capacity compared with the scaled loss can be unstable.

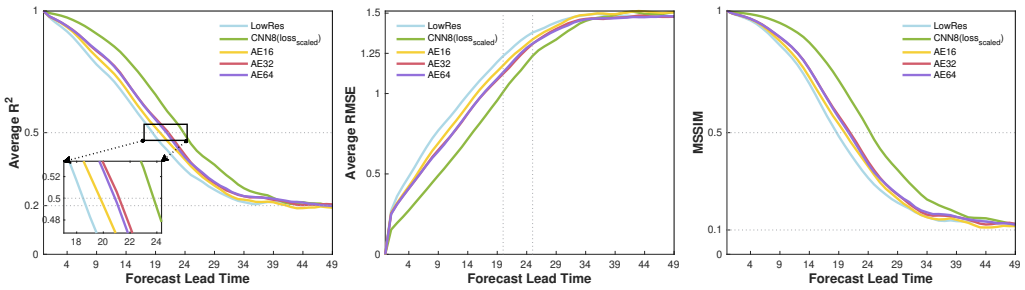

Figure 4. Curves of averaged R^2, RMSE and MSSIM over time for the AE model with different look-ahead-steps, where R is the correlation coefficient and MSSIM is the multi-scale structural similarity.

Simulation	Forecast lead time	Computational time (total)	DL time	Forecast lead time improvement		Total time saved	Time ratio between DL and dynamical core
				Compared to LowRes	Compared to CNN8(loss_{scaled})		
LowRes	19.48	1.01	N / A	N / A	N / A	N / A	N / A
CNN8(loss_{scaled})	24.76	16.42	15.41	27.09 %	N / A	N / A	15.26
CNNhk32	25.80	6.65	5.64	32.42 %	4.20 %	59.52 %	5.58
CNN32(SGM)	25.86	11.22	10.21	32.73 %	4.44 %	31.66 %	10.12
CNN32(CFS)	25.57	7.69	6.68	31.22 %	3.25 %	53.15 %	6.62

Table 1. Forecast lead time and computational time consumption for different models.

Due to the outstanding and steady performance of the scaled loss function, it is chosen to be applied for all refined models and CNN8(loss_{scaled}) will be regarded as the baseline for comparisons. All the experiments are carried out and compared using a 3090 ti GPU. Figure 3b shows the performance of all the refined models. CNNhk32 has a similar performance with CNN32 applied with SGM, which is slightly better than CNN32 using CFS. Table 1 shows the precise data of extended forecast lead time, the computational time, and comparative analysis between the refined models and the baseline model. By applying 32 look-ahead-steps for the refined models, on one hand, they can all attain the same and even a little bit higher level of model forecast performance compared with the initial model with 8 look-ahead-steps, which effectively extends the forecast lead time to around 30% than the LowRes simulation. And on the other hand, all the refined models can significantly reduce computational time cost, from around 30% to 60%, and the ratio of time cost between DL inference and dynamical core can be reduced from around 15 to 5.58. That is to say, by training with longer look-ahead-steps, these three methods can all work for improving the initial model efficiency without any loss of the model capacity.

4 Summary and Discussion

To conclude, in this study we aim to enhance our initial CNN model's efficiency for the parameterization of the SGS process and wish to reach a minimum loss of extended forecast lead time. To realize this purpose, we design an ideal case with BVE model and employ the SOL method. By containing longer look-ahead-steps into the training process of our newly-designed three models (CNNhk, SGM and CFS), similar model capacities can be obtained with computational time reduced by 31.66% to 59.52% separately, with dataset generated from the simulation of the BVE model. Overall, our work technically provides some promising approaches for improving DL models' efficiency when they are applied for scientific tasks where in-time forecast results are necessarily needed, and through this study, we expect to further inspire more investigations and explorations.

According to the experiment results, the CFS is only a little bit worse than SGM, but it spends obviously less computational time. Initially, this implies the great potential for making maximum use of information in the key areas, which will significantly reduce the amount of training data but still keep the model's capacity. But does this factor negate the practicality of SGM? Firstly, the initial domain used in this study is small, which only contains 4096 grids, thus the amount of parameters used for the coarse part of SGM accordingly accounts for a small proportion in the total amount of parameters of the model, as a result of which the difference in forecast lead time improvement between SGM and CFS is not so obvious. But if a larger initial domain is applied, SGM is expected to perform much better than CFS. Secondly, the insignificant computational time reduction for SGM is due to the technical reason, where the operation for downscaling the original domain is needed for every time of integration. This problem can be solved with future development of computer algorithm, and overall SGM is thought as a more reliable and stable method than CFS.

Moreover, longer look-ahead-steps are used to keep the refined models' capacity in this study, but does it mean that longer look-ahead-steps can be used to improve the model forecast ability without a limit and thus any compact model all can be used for improving the initial model efficiency? The autoencoder (AE) models are famous for its compact architecture and have been widely used for data reconstructions (Y. Wang et al., 2022), so it is also tested (Figure 4). But because generating a latent vector seemingly will lead to a great information loss of the input data, the tested AE model performs even worse than the baseline (CNN8(loss_{scaled})). When we try to include more look-ahead-steps into the AE model to improve its forecast ability, it is found that there exists a limit for look-ahead-steps. When we adding look-ahead-steps from 16 to 32, forecast lead time improvement is observed. But when we add it up to 64 look-ahead-steps, no obvious enhancement but even deterioration appears, which implies there exists a maximum number of the look-ahead-steps for DL models. What's more, the AE model with 32 and 64 look-ahead-steps is still worse than the baseline, which further denotes that not all the compact model are suitable for this learning task.

Another interesting phenomenon observed is that the status of gradient descent cannot always actually represent the final model capacity. We test various checkpoints saved within the training process, and it turns out that a latter checkpoint with smaller training and validation loss sometimes can be worse than a former one. This does not constitute an overfitting issue as the validation loss continues to decrease, but it should have something to do with the inconsistency between the pretrained dynamical core and the actual one when we combine PDEs with DL models. This can be further evidenced by the experiment facts of a pretrained DL model with only one look-ahead-step, which contains no interaction between PDEs and DL models, that normal gradient descent is also observed during the training process but the pretrained model will perform worse than LowRes during online tests. The PDE is not trained in this experiment but there is nothing wrong with the gradient descent of the loss function, which implies that the status of the loss function does not represent the actual state of trained PDEs. Similar problems have also been reported in previous work (Krishnapriyan et al., 2021; S. Wang et al., 2021; C. Wang et al., 2022), which indicates that there still exist some unknown mechanism when training PDEs in DL models. And for the future work, we will continue to explore the design for loss functions or to find an evaluation standard that can better represent the exact training states for PDEs.

References

Alapaty, K., Herwehe, J. A., Otte, T. L., Nolte, C. G., Bullock, O. R., Mallard, M. S., ... Dudhia, J. (2012). Introducing subgrid-scale cloud feedbacks to radiation for regional meteorological and climate modeling. *Geophysical Research Letters*, *39*(24). doi: https://doi.org/10.1029/2012GL054031

Bi, K., Xie, L., Zhang, H., Chen, X., Gu, X., & Tian, Q. (2023). Accurate medium-

range global weather forecasting with 3d neural networks. *Nature*, *619*, 533–538. doi: https://doi.org/10.1038/s41586-023-06185-3

Bradbury, J., Frostig, R., Hawkins, P., Johnson, M. J., Leary, C., Maclaurin, D., ... Zhang, Q. (2018). *JAX: composable transformations of Python+NumPy programs.* Retrieved from `http://github.com/google/jax`

Chen, K., Han, T., Gong, J., Bai, L., Ling, F., Luo, J.-J., ... Ouyang, W. (2023). Fengwu: Pushing the skillful global medium-range weather forecast beyond 10 days lead.

Chollet, F. (2017). *Xception: Deep learning with depthwise separable convolutions.*

Cuomo, S., Cola, V. S. D., Giampaolo, F., Rozza, G., Raissi, M., & Piccialli, F. (2022). Scientific machine learning through physics–informed neural networks: Where we are and what's next. *Journal of Scientific Computing*, *92*(88). doi: https://doi.org/10.1007/s10915-022-01939-z

Goger, B., Rotach, M. W., Gohm, A., Stiperski, I., & Fuhrer, O. (2016). Current challenges for numerical weather prediction in complex terrain: Topography representation and parameterizations. In *2016 international conference on high performance computing simulation (hpcs)* (p. 890-894). doi: 10.1109/HPCSim.2016.7568428

Guo, Y., Li, Y., Feris, R., Wang, L., & Rosing, T. (2019). *Depthwise convolution is all you need for learning multiple visual domains.*

Kochkov, D., Smith, J. A., Alieva, A., Wang, Q., Brenner, M. P., & Hoyer, S. (2021). Machine learning–accelerated computational fluid dynamics. *Proceedings of the National Academy of Sciences*, *118*(21), e2101784118. doi: 10.1073/pnas.2101784118

Kovachki, N. B., Li, Z., Liu, B., Azizzadenesheli, K., Bhattacharya, K., Stuart, A. M., & Anandkumar, A. (2021). Neural operator: Learning maps between function spaces. *CoRR*, *abs/2108.08481*. Retrieved from `https://arxiv.org/abs/2108.08481`

Krishnapriyan, A. S., Gholami, A., Zhe, S., Kirby, R. M., & Mahoney, M. W. (2021). *Characterizing possible failure modes in physics-informed neural networks.*

Lam, R., Sanchez-Gonzalez, A., Willson, M., Wirnsberger, P., Fortunato, M., Pritzel, A., ... Battaglia, P. (2022). Graphcast: Learning skillful medium-range global weather forecasting.

Lu, L., Jin, P., Pang, G., Zhang, Z., & Karniadakis, G. E. (2021). Learning nonlinear operators via deeponet based on the universal approximation theorem of operators. *Nature Machine Intelligence*, *3*, 218-229. doi: https://doi.org/10.1038/s42256-021-00302-5

Mantovani Júnior, J. A., Aravéquia, J. A., Carneiro, R. G., & Fisch, G. (2023). Evaluation of pbl parameterization schemes in wrf model predictions during the dry season of the central amazon basin. *Atmosphere*, *14*(5). doi: https://doi.org/10.3390/atmos14050850

Müller, M., & Scherer, D. (2005, 06). A grid and subgrid-scale radiation parameterization of topographic effects for mesoscale weather forecast models. *Monthly Weather Review - MON WEATHER REV*, *133*, 1431-1442. doi: 10.1175/MWR2927.1

Qu, Y., & Shi, X. (2023). Can a machine learning–enabled numerical model help extend effective forecast range through consistently trained subgrid-scale models? *Artificial Intelligence for the Earth Systems*, *2*(1), e220050. doi: https://doi.org/10.1175/AIES-D-22-0050.1

Raissi, M., Perdikaris, P., & Karniadakis, G. (2019). Physics-informed neural networks: A deep learning framework for solving forward and inverse problems involving nonlinear partial differential equations. *Journal of Computational Physics*, *378*, 686-707. doi: https://doi.org/10.1016/j.jcp.2018.10.045

Sherwood, S., Bony, S., & Dufresne, J. L. (2014). Spread in model climate sensitivity traced to atmospheric convective mixing. *Nature*, *505*, 37–42. doi: https://doi.org/10.1038/nature12829

Shi, X. (2020). Enabling smart dynamical downscaling of extreme precipitation events with machine learning. *Geophysical Research Letter*, *47*(19), e2020GL090309. doi: https://doi.org/10.1029/2020GL090309

Um, K., Brand, R., Fei, Y., Holl, P., & Thuerey, N. (2020). Solver-in-the-Loop: Learning

from Differentiable Physics to Interact with Iterative PDE-Solvers. *Advances in Neural Information Processing Systems*.

Wang, C., Li, S., He, D., & Wang, L. (2022). *Is l^2 physics-informed loss always suitable for training physics-informed neural network?*

Wang, S., Teng, Y., & Perdikaris, P. (2021). Understanding and mitigating gradient flow pathologies in physics-informed neural networks. *SIAM Journal on Scientific Computing*, *43*(5), A3055-A3081. Retrieved from `https://doi.org/10.1137/20M1318043` doi: 10.1137/20M1318043

Wang, Y., Shi, X., Lei, L., & Fung, J. C.-H. (2022). Deep learning augmented data assimilation: Reconstructing missing information with convolutional autoencoders. *Monthly Weather Review*, *150*(8), 1977 - 1991. Retrieved from `https://journals.ametsoc.org/view/journals/mwre/150/8/MWR-D-21-0288.1.xml` doi: https://doi.org/10.1175/MWR-D-21-0288.1

Wang, Z., Simoncelli, E., & Bovik, A. (2003). Multiscale structural similarity for image quality assessment. In *The thrity-seventh asilomar conference on signals, systems computers, 2003* (Vol. 2, p. 1398-1402 Vol.2). doi: 10.1109/ACSSC.2003.1292216