Neural Network Emulator Optimizer: A Preliminary Study on Korean Microphysics Parameterization Model*

Sojung An[0000-0002-0170-1031], Inchae Na, Tae-Jin Oh**, and Junghan Kim

Korea Institute of Atmospheric Prediction Systems
35, Boramae-ro, Dongjak-gu, Seoul, Korea
{sojungan, icna, oht, jhkim}@kiaps.org

Abstract. Recent years have witnessed great progress in emulators based on neural network (NN). Current state-of-the-art emulators methods often apply shallow NN to attain high performance in physics system, which brings a faster speed processing on resource-constrained environments. Although several works have focused on improving accuracy in physics emulators, an effective and efficient method for tackling time consuming problem of existing system on high-resolution remains lacking. In this paper, we propose a optimum NN emulator of a microphysics (MPS) parameterization scheme to effectively solve the problem in numerical weather prediction (NWP) model, Korea Integrated Model (KIM) in particular. Specifically, we adopt a shallow NN to build an intelligent emulator, which can learn the feature map and estimate the vertical MPS forcing increment profiles. This study mainly relies on two technical contributions: (1) Optimization: reviewing and improving models for simulating non-linear parameters; (2) Feasibility: efficient computation with minimal loss of physical information. We validate the proposed model with four seasons (10-day forecast at 200 seconds interval) of KIM. Results indicate that the proposed single-layer network shows best performance for emulating MPS in KIM. Our analyses will provide a guideline for optimal physical parameterizations modeling.

Keywords: Emulator · Microphysics · Physical Parameterization · Neural Network · Feature Extraction

1 Introduction

In recent decades, there have been considerable improvements in terms of predictability in numerical weather prediction (NWP) models. Advancement in computer hardware technology is considered one of the big drivers accounting for improvement in this area. However, increase in spatial-temporal resolution for NWP models, which is critical for better predictability, requires exponential increase of computational power. Thus, utilizing even more computational resource and/or achieving better model code optimization is continually

* Copyright © by the paper's authors. Use permitted under Creative Commons License Attribution 4.0 International (CC BY 4.0). In: N. D. Vo, O.-J. Lee, K.-H. N. Bui, H. G. Lim, H.-J. Jeon, P.-M. Nguyen, B. Q. Tuyen, J.-T. Kim, J. J. Jung, T. A. Vo (eds.): Proceedings of the 2nd International Conference on Human-centered Artificial Intelligence (Computing4Human 2021), Da Nang, Viet Nam, 28-October-2021, published at http://ceur-ws.org

** Corresponding author.

needed. Parameterization of subgrid-scale physical processes has been an active research area ever since the birth of NWP. There are extensive research activities on physics parameterization [3, 5, 6, 9, 13] utilizing Korea Integrated Model (KIM), currently the operational NWP model in Korea, which is built upon non-hydrostatic governing equations and discretized with spectral element and finite difference method in horizontal and vertical dimensions, respectively [4]. The physics module parameterizes subgrid-scale physical processes which cannot be resolved in the grid-scale in NWP model. Subgrid-scale physics parameterization development in most cases relies on observational data which are parameterized to make it fit the observation. Also, physics parameterization is computed in vertical columns in terms of the three-dimensional data grid structure and is agnostic to the neighboring horizontal grids. These properties make machine learning an ideal tool for developing physics parameterization as approximating complex nonlinear mappings can be done effectively with machine learning, e.g., neural networks..

Previous studies of parameterization based on machine learning can be categorized into developing *emulation-based* methods for accelerated calculation [7, 8, 10–12, 16, 17] and developing new *empirical* parameterizations based on observation or high resolution model data [1]. Recently, a shallow Neural Network emulator which covers the entire suite of physical parameterization has been developed [1] which is based on a single hidden layer. Their model showed satisfactory accuracy with a much faster execution in simulating nonlinear physics modules compared to the original code. By exploiting faster computation of the NN based emulator approach, NWP models can be run effectively on a higher resolution. There are competent studies using NN as emulators (e.g., Krasnopolsky et al., Nadiga et al.) but studies regarding optimization of network architecture are yet to be found. When artificial intelligence recognizes patterns in physics, the structure of neurons, depth of layers, and the choice of activation functions are important factors. For example, the Rectified Linear Unit (ReLU) given by max(0, x) is the most often used activation function in the deep learning community. However, when it comes to physics-informed neural networks, ReLU is reported to result in spurious wiggles in the computed derivatives representing fluid flows [15]. Thus, understanding the underlying physics of your target system in detail is critical for effective network design.

In this study, we explore various types of NN structure that represent physics of the atmosphere. Specifically, we compare a series of NN models to test its effectiveness with the combination of the following options: (i) numbers of each layer, (ii) neuron structure, and (iii) activation function.

2 Related Work

This section describes several studies based on emulation similar to our proposed model. We classify these studies into two groups: (i) *single-physics* and (ii) *unified-physics* emulation.

2.1 Single-physics emulations

The state-of-the-art models for physics emulator include a step that optimizes hidden layer based on machine learning, or approximates these weights based on various activation functions. Many previous studies demonstrated that the NN emulator approach can be applied successfully to speed up the computation of a single physics module. Machine learning is first used to emulate longwave radiation for the European Centre for Medium-Range Weather Forecasts models [2]. Krasnopolsky reduced computation time by one to two orders of magnitude of decadal climate simulations [7, 8]. Also, the authors verified that the speed of longwave radiation emulators based on NN can be increased by 50 to 80 times [11].

Aside from these, multi-perceptrons have been proposed for predicting nonlinear phenomena in the physics of the atmosphere influenced by a number of factors. To accelerate expensive radiative transfer computations, deep NN has been applied to predict vertical profiles of longwave and shortwave radiative fluxes in weather and climate models [12]. Veerman developed a parametrization to emulate a radiation parametrization based on multi-perceptron and leaky relu [17]. Roh evaluated the forecast performance of radiation emulators with 300 to 56 neurons with sigmoid activation for cloud-resolving simulation [16]. The Emulators surpassed the speed of the existing model in a single physics process. However, when integrated with the main NWP model, the speed up effect was limited [10].

2.2 An unified-physics emulation

Traditional emulation-based methods were parameterized based on NN by dividing the physical process separately. Some errors arising from these models had a significant effect on the accuracy of the overall NWP model. As all physical processes closely interacts with each another, one error source in a particular physics module can cause larger errors in other sub-physics. Belochitski proposed a shallow NN based emulator of a complete suite of atmospheric physics parameterizations [1]. The paper aims to learn all domains of physics intimately connected. These methods learn an encoder to extract the physical features, and maintain the representation consistency through minimizing the reconstruction error between all physics. Their implementation can handle long-term numerical integration as well as providing 3 times faster computation than the original physics module. However, these advantages come with a huge computational cost, and it is hard to train high-resolution data. Fortunately, the paper achieved good results at a high resolution of 25km, despite their model being trained with a 100km-resolution source data.

3 A Network for Emulation-based Parameterizations

This paper studies the problem of physics from the perspective of a NN to propose an ***emulation-based*** algorithm.

3.1 Defining the Weighted Matrix

Physics emulator is to learn input features on a network to map the variables into next input. Let $X(t)$ and $Y(t)$ represent the input and output at time t, respectively. Denote $X = (x_0, x_1, \cdots, x_n)^T \in \mathbb{R}^n$as the input information related to physics observed on the network and $Y \in \mathbb{R}^m$ as the output information, where n and m are the number of input and output dimension, respectively. The purpose of training is to find θ that maximizes the conditional probability of input-output pairs in the training sets. On each t step, the decoder receives the features of the previous input. If the model is trained to be predicting the output with constantly updated input presented in the previous phase, we are training function Φ successively as:

$$\Phi = \left[x_0^{(t)}, \cdots, x_n^{(t)}; \theta\right] \rightarrow \left[x_0^{(t+1)}, \cdots, x_m^{(t+1)}; \theta\right], \tag{1}$$

where t denotes time step. Inputs and outputs consist of the same attributes for each step and each output attributes are connected in a recursive manner. For any x_i, the inputs are calculated according to the following formula:

$$x' = \begin{cases} 1, & \text{if } x > max \\ \dfrac{x - min}{max - min}, & \text{if } min \leq x < max \\ 0, & \text{otherwise} \end{cases} \tag{2}$$

The min and max range of each attribute are set manually as physically allowable values in KIM. We define the emulator network as a weight vector W representing the state of the entire attributes and concatenate each value associated variables as W. $W \in \mathbb{R}^{n \times m}$ represents the weighted matrix:

$$W = \begin{bmatrix} w_{1,1} & w_{1,2} & \cdots & w_{1,n} \\ w_{2,1} & w_{2,2} & \cdots & w_{2,m} \\ \vdots & \vdots & \ddots & \vdots \\ w_{n,1} & w_{n,2} & \cdots & w_{n,m} \end{bmatrix}_{n \times m}, \tag{3}$$

where $w_{i,j}$ is the weights between input x_i and output y_j of physics.

3.2 Parameterization Network of the Physical Feature Extraction

Since the network is a matrix, for any value $w_{i,j}$ in the matrix, its output y_i is defined as the following:

$$y_i = \sum_{i=1}^{M} x'_i w_{i,j} + b_j. \tag{4}$$

Using the definition of layers, we can obtain the output matrix of the size $n \times m$. Assume that there are L network layers, where the 0^{th} layer is the input layer and the $l^{th} (1 \leq l \leq L)$ layers are fully connected layers. For any fully connected layer $l \in [1, \cdots, L]$, the output of the $l^{th}(1 \leq l \leq L)$ fully connected layer is calculated using the following equation:

$$H_r^{(l)} = \sigma(\sum_{s=1}^{s_{l-1}} f_\theta^l H_s^{(l-1)}), \tag{5}$$

where s_{l-1} denotes the number of parameters in the $(l-1)^{th}$ layer; $H_s^{(l)} \in \mathbb{R}^{s_{l-1}}$ is the s^{th} physics hidden parameter in the l^{th} layer and $\sigma(\cdot)$ is the activation function. The neuron structure can be divided into four following structures according to the change in the number of each neuron:

$$\begin{aligned} \alpha &: s_L < \{s_2, \cdots, s_{l-1}\} \leq s_1 \\ \beta &: s_L < s_1 < \{s_2, \cdots, s_{l-1}\} \\ \gamma &: \{s_2, \cdots, s_{l-1}\} < s_L < s_1 \\ \delta &: s_L \leq \{s_2, \cdots, s_{l-1}\} < s_1, \end{aligned} \tag{6}$$

where $\{S_2, \cdots, S_{l-1}\}$ denotes the number set of hidden neurons connected to input layer $H_s^{(1)}$, and s_l is a number of output neurons. Activation functions

Table 1: The mechanisms of five activation approaches

Types	Structures of activation function *	References
Hyperbolic tangent	$\frac{e^\theta - e^{-\theta}}{e^\theta + e^{-\theta}}$	[1,7,8,12]
Leaky ReLU	$\begin{cases} \theta, & \text{if } \theta > 0 \\ 0.01\theta, & \text{otherwise} \end{cases}$	[17]
Sigmoid	$\frac{1}{1+e^{-\theta}}$	[16]
Swish	$\theta \cdot sigmoid$	-

* Throughout the table, $\sigma(\theta)$ is the activation function of parameter θ.

tested in this study are described in Table 1. The s_{l-1}, number of l, and $\sigma(\cdot)$, defined in this section, are chosen as optimized formula by experiments in the next section. Finally, the errors between actual scores and the predicted scores are minimized by *L1-norm*.

4 Evaluation

In this section, we evaluate the NN model for emulating physics among the models proposed in section 3.

4.1 Datasets and Implementation Details

To verify the efficacy of the proposed methods, we conducted experiments with generated MPS dataset for every 200 seconds using ERA5 reanalysis by latest KIM version 3.6. MPS has been selected for this first study because the nonlinearity of MPS is hard to predict and it is one of the most time-consuming part of the atmospheric physical processes. The input and output attributes are shown in Table 2. The dataset consists of 4 sets with each season to consider the temporal features. Dynamical core of the NWP is a spectral element cubed-sphere nonhydrostatic model with a horizontally quasi-uniform resolution of 100km, with 91 vertical layers (˜0.01hpa) on a hybrid sigma-pressure vertical coordinate. Data is extracted randomly in each KIM forecasting data, and composed of 6,998,688 training sets. We use the other 1,749,672 sets for evaluation. Our methods are implemented using PyTorch [14] and optimized using Adam ($\lambda = 10^3$). For fairness, we train all networks with a batch size of 128 for 500 epochs, on random sets.

Table 2: Summary of datasets used in MPS. The data consists of 822 dimension input and 548 dimension output. Output, a part of the inputs, is changed every time-step in MPS.

	Input Layers		Output Layers	
Attribute	# of index	# of profile	# of index	# of profile
Grid distance	1	1	-	-
Precipitation	2	1	1	1
Snow	3	1	2	1
Pressure	4:94	1:91	-	-
Depth of layer	95:184	1:91	-	-
Virtual temperature	186:276	1:91	-	-
Temperature	277:367	1:91	3:93	1:91
Specific humidity	368:458	1:91	94:184	1:91
Mixing ratio *	459:822	1:364	185:548	1:364

* Mixing ratio of cloud, rain, ice, and snow

We evaluate our emulation both quantitatively and qualitatively for estimating the optimal emulator. The results of this experiment are evaluated with three metrics shown in Eq. (7): Root Mean Square Error (RMSE), mean absolute error (MAE), and Peak Signal-to-noise ratio (PSNR). Given NN output to e_i and KIM output to $\bar{e_i}$, the evaluation function can be written as:

$$\begin{aligned} MAE &= \frac{1}{n}\sum_{i=1}^{n}|e_i - \bar{e_i}| \\ RMSE &= \sqrt{\frac{1}{n}\sum_{i=1}^{n}(e_i - \bar{e_i})^2} \\ PSNR &= 10 \cdot \log(\frac{max^2}{n}\sum_{i=1}^{n}(e_i - \bar{e_i})^2), \end{aligned} \tag{7}$$

4.2 Case Analysis

For comparison, we set a benchmark case that consists of 2-layers NN based on a hyperbolic tangent and the neuron structure of γ. Table. 3 depicts the average accuracy of emulations with different approaches. In our first experiment, we

Table 3: MPS output result between KIM and NN emulator

	Network Structure (S)				Depth of Layer (L)				Ativation Function (σ)			
	α	β	γ	δ	1	2	3	4	*tanh*	*relu*	*sigmoid*	*swish*
MAE (E-04)	2.46	1.44	3.49	2.08	0.74	2.08	5.22	7.45	2.08	3.67	8.68	2.14
RMSE (E-04)	9.96	0.67	12.27	7.71	1.72	7.71	17.52	24.81	7.71	11.74	25.53	7.79
PSNR	30.56	32.61	29.55	31.83	38.39	31.83	27.96	26.41	31.83	29.66	26.14	31.72

* α, β, γ, and δ are structures of neuron described in section 3.

evaluate the impact of learning each structure for estimating the optimal depth of NN layers. As L is 2 in the benchmark case, we set the hidden neurons to $\alpha_{s_2} \rightarrow 822$, $\beta_{s_2} \rightarrow 1096$, $\gamma_{s_2} \rightarrow 400$, and $\delta_{s_2} \rightarrow 548$, respectively. The PSNR result indicates that the performances of emulation approaches by applying the β method increased from 29.55 (the γ method commonly used in emulation) to 32.61 of the average accuracy of emulation. The β structure also shows overall best performance compared to other network structures as shown in the evaluation metric scores. When the number of hidden neurons decreases relative to the input, such as in the γ structure, loss of latent features is inevitable. These losses can cause uncertainties of physics, so the hidden neurons must be greater than or equal to the number of output neurons.

Let us compare the impact of learning according to layer depth and activation function. All networks to which the deep layer was applied except for the single-layer applied the fully connected layer, and then proceeded with setting the activation function and dropout (0.2). The result of approximation errors presented above shows that the shallow NN is capable of providing NN emulations with small error. Especially, single-layer yields best performance compared to other cases. As for the activation, the Leaky ReLU with a bend in the slope shows the worst performance, consistent with previous studies. The results demonstrate that hyperbolic tangent activation for emulating MPS is effective, providing accurate and visually promising results. The swish activation function operated well as hyperbolic tangent with $2.14(\cdot 10^{-4})$ error. Generally, good results are obtained from functions with a soft and high gradient.

Finally, we compare the original KIM MPS output with the NN emulator output. Fig. 1 illustrates the correlations between outputs from the NN-based emulation with the outputs of the MPS. The single-layer-based emulator, which is the best result in Table 3, was used to emulate the output of the MPS. The figure is outputs according to 5-day input data integrated by KIM in consideration of spin-up.

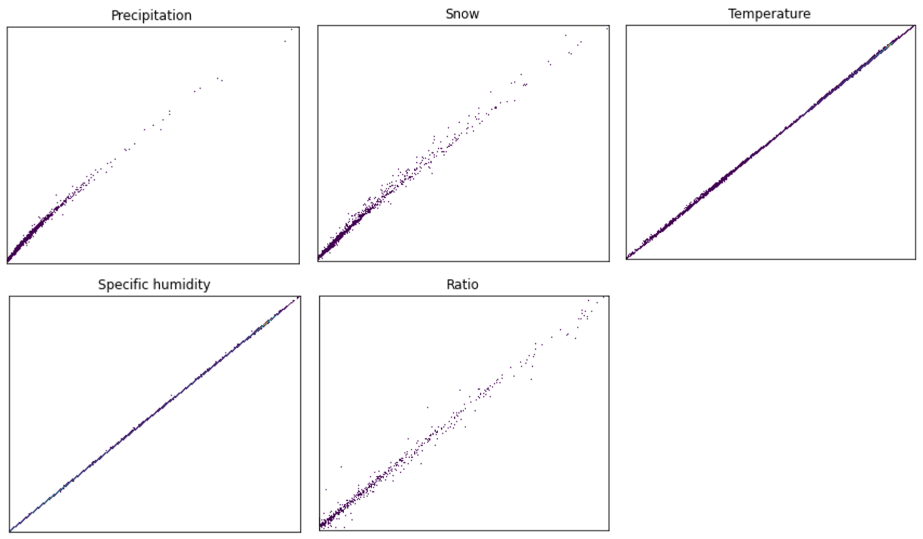

Fig. 1: Scatter plot (physics output vs. NN output) for output attributes. The scatter plot compares the first profile of the attributes shown in Table 2. The x-axis is physics and the y-axis is the NN value.

Average of Correlation coefficients is 0.998 (Precipitation: 0.989, Snow: 0.989, Temperature: 0.999, Specific humidity: 0.999, and Ratio: 997). Emulators with shallow NN also verified similar results. KIM MPS based, NN emulation, and their difference in precipitation distribution is shown in Fig. 2. Precipitation simulations of KIM and NN are shown in (a) and (b); (c) is the difference of the simulations, with single-layer emulator. The precipitation distributions for the KIM and NN emulator runs are very similar showing little difference as shown in (c) in Fig. 2.

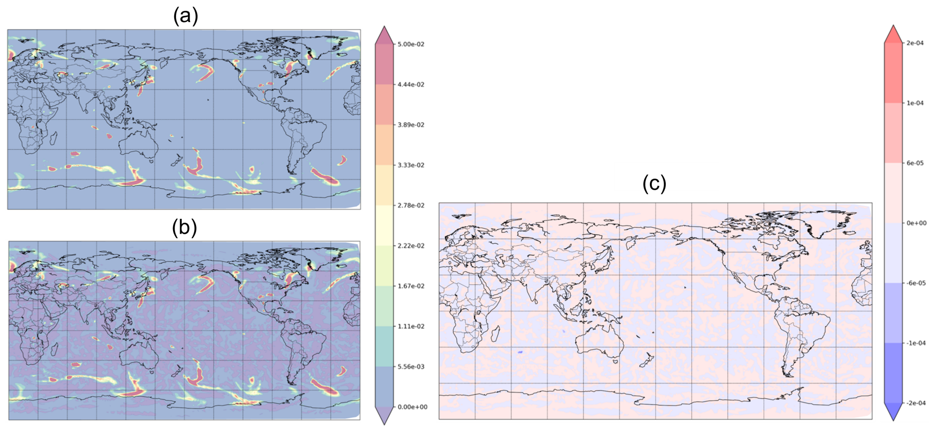

Fig. 2: The output of precipitation and the difference between the output and MPS precipitation output. The range of the color bar on the left is 0 to $5(\cdot 10^{-2})$, and the color bar on the other side is $-1.8(\cdot 10^{-4})$ to $1.8(\cdot 10^{-4})$.

5 Conclusion and Future Work

As information technology evolves, the tasks of accelerating physics parameterization and increasing the resolution of NWP have been gaining importance. Traditional emulator methods tend to design only on the overall composition such as a kind of input data. This paper focused on building an optimal NN targeted according to the detail of network settings for physics emulation. Various experiments were carried out to design the detailed structure of the network, and we tried to design a network that can understand the characteristics of the physics. Specifically, to overcome the drawbacks of existing models, we have attempted methods for optimizing NN details (i.e., neuron structure, number of layers, and activation function). Through experiments parameterization of the MPS using a single layer showed the best performance.

In the case of MPS, the usage of deep-layer does not bring the best result but it may bring great results in other physics parameterizations. The depth of layer can be changed according to physics patterns and other elements (e.g. number of profiles, resolution, parameterization type, so on) used by the NWP model. So we want an emulation that feasible not only single physics, but also across physics module. Ideally, we will achieve unified-physics emulation by considering different physical patterns, leveraging dynamic NN modeling.

Acknowledgement

This work was carried out through the R&D project ”Development of the Next-generation Operational System of Korea Institute of Atmospheric Prediction Systems (KIAPS)”, funded by Korea Meteorological Administration (KMA2020-02213).

References

1. Belochitski, A., and Krasnopolsky, V.: Stable Emulation of an Entire Suite of Model Physics in a State-of-the-Art GCM using a Neural Network, arXiv preprint arXiv:2103.07028, (2021).
2. Chevallier, F., Chéruy, F., Scott, N. A., and Chédin, A.: A neural network approach for a fast and accurate computation of a longwave radiative budget. Journal of applied meteorology, **37**(11), 1385-1397 (1998).
3. Han, J. Y., Hong, S. Y., Sunny Lim, K. S., and Han, J.: Sensitivity of a cumulus parameterization scheme to precipitation production representation and its impact on a heavy rain event over Korea. Monthly Weather Review, **144**(6), 2125-2135 (2016). 144, 2125-2135
4. Hong, S. Y. et al.,: The Korean Integrated Model (KIM) system for global weather forecasting. Asia-Pacific Journal of Atmospheric Sciences, **54**(1), 267-292 (2018)
5. Kim, E. J., and Hong, S. Y.: Impact of air-sea interaction on East Asian summer monsoon climate in WRF. Journal of Geophysical Research: Atmospheres, **115**(D19) (2010).
6. Koo, M. S., Choi, H. J., and Han, J. Y.: A parameterization of turbulent-scale and mesoscale orographic drag in a global atmospheric model. Journal of Geophysical Research: Atmospheres, **123**(16), 8400-8417 (2018).

7. Krasnopolsky, V. M., Fox-Rabinovitz, M. S., and Chalikov, D. V.: New Approach to Calculation of Atmospheric Model Physics: Accurate and Fast Neural Network Emulation of Longwave Radiation in a Climate Model, Monthly Weather Review **133**(5), 1370–1383 (2005).
8. Krasnopolsky, V. M., Fox-Rabinovitz, M. S., and Belochitski, A. A.: Decadal Climate Simulations Using Accurate and Fast Neural Network Emulation of Full, Longwave and Shortwave, Radiation, Monthly Weather Review, **136**(10), 3683–3695 (2008).
9. Lee, E. H., Lee, E., Park, R., Kwon, Y. C., and Hong, S. Y.: Impact of turbulent mixing in the stratocumulus-topped boundary layer on numerical weather prediction. Asia-Pacific Journal of Atmospheric Sciences, **54**(1), 371-384 (2018).
10. Morcrette, J. J., Mozdzynski, G., and Leutbecher, M.: A reduced radiation grid for the ECMWF Integrated Forecasting System, Monthly weather review, **136**(12), 4760-4772 (2008).
11. Nadiga, S., Krasnopolsky, V., Bayler, E. J., Mehra, A., Kim, H. C., and Behringer, D.: Neural Network Technique For:(a) Gap-Filling Of Satellite Ocean Color Observations, And (b) Bridging Multiple Satellite Ocean Color Missions. In AGU Fall Meeting Abstracts, **2015**, IN43C-1755 (2015).
12. Pal, A., Mahajan, S., and Norman, M. R.: Using deep neural networks as cost-effective surrogate models for super-parameterized E3SM radiative transfe,. Geophysical Research Letters, **46**(11), 6069-6079 (2019).
13. Park, R. S., Chae, J. H., and Hong, S. Y.: A revised prognostic cloud fraction scheme in a global forecasting system. Monthly Weather Review, **144**(3), 1219-1229 (2016).
14. Paszke, A., Gross, S., Massa, F., Lerer, A., Bradbury, J., Chanan, G., and Chintala, S.: Pytorch: An imperative style, high-performance deep learning library, Advances in neural information processing systems, 32, 8026-8037 (2019).
15. Raissi, M., Yazdani, A., and Karniadakis, G. E.: Hidden fluid mechanics: Learning velocity and pressure fields from flow visualizations. Science, **367**(6481), 1026-1030 (2020).
16. Roh, S., and Song, H. J.: Evaluation of neural network emulations for radiation parameterization in cloud resolving model. Geophysical Research Letters, **47**(21), e2020GL089444 (2020).
17. Veerman, M. A., Pincus, R., Stoffer, R., Van Leeuwen, C. M., Podareanu, D., and Van Heerwaarden, C. C.: Predicting atmospheric optical properties for radiative transfer computations using neural networks. Philosophical Transactions of the Royal Society A, **379**(2194), 20200095 (2021).