SULI Technical Report

SUMMER 2021

Emulation of the Kessler Microphysics Scheme with Deep Neural Networks

Zane Bayer

Slippery Rock University of Pennsylvania

Supervised by
Valentine Anantharaj
National Center for Computational Sciences

In collaboration with
Matthew Norman
National Center for Computational Sciences

Submitted: August 2nd, 2021

ABSTRACT

Numerical weather prediction (NWP) models are now used extensively in weather prediction. NWP models are comprised of many algorithms with varying computational costs. Scientists are exploring methods to improve the performance of computationally intensive routines within their models. Over the last decade, research has focused on improving computational capacity, physics algorithms, and data assimilation methods, especially for satellite-based sensor data. Deep learning methods are being actively explored in the context of NWP models. Here, we investigated the applicability of deep neural networks as emulators within these weather models, specifically the Kessler microphysics (KMP) scheme used in NWP models to condense water vapor into rain. We used machine learning models to emulate the KMP scheme. We collected data from the input and output to the Kessler microphysics subroutine in a cloud model and trained two sets of machine-learning (ML) models for classification and regression. Numerical methods for parameterizations are subject to machine precision in calculations. In some instances, the noise due to machine precision tend to contaminate and confuse machine learning models. So, our approach is to first train a ML model for classifying data samples susceptible to machine precision noise; removing those samples; and then train another ML model to emulate the KMP scheme. We also provide logistic and decision tree regression baselines for comparison. These baselines allowed us to gauge the potential for improving the performance of the KMP scheme with deep neural networks. Our research showed that machine precision needs to be considered when using data from simulation outputs for developing ML emulators.

I. INTRODUCTION

The use of deep learning, a subtopic of machine learning (ML) based on the use of many layered artificial neural networks, has proved valuable in the natural sciences. These deep learning techniques have proved particularly useful for analysis of high-dimensional scientific data[1,2]. Recently, deep learning research has explored the use of deep neural networks (DNNs) in weather forecasting[3]. Within this domain, NWP models provide reliable guidance for weather forecasting, but comes at a high computational cost. Researchers have been exploring the use of DNNs to lower the computational cost of NWP. While a full replacement of numeric weather models remains out of reach, DNNs may prove useful in emulating costly schemes[4].

Here, we focus on the emulation of the Kessler Microphysics scheme (KMP) with DNNs. The KMP scheme is an appealing candidate to test DNN emulators due to its simplicity and widespread availability in NWP models. The KMP scheme is a parameterization for calculating the amount of rain (if any) in a grid cell based on the composition of moisture (mixing ratios of water vapor, liquid water and rain water). The KMP scheme accounts for the condensation of cloud water from water vapor, evaporation of rain and cloud water, the generation of rain and the falling out of rain.

The moisture composition is controlled by updating three values which indicate the amount of cloud water, water vapor, and liquid rain. Cloud microphysics plays a vital role in a NWP model as it can have a drastic effect on other parameters. For instance, the amount of radiation absorbed, reflected, and trapped in the atmosphere is highly dependent on the makeup of the clouds. For a

more detailed description of the KMP scheme, see reference 5. Due to its vital role and widespread use, the KMP scheme is a good initial test for DNN emulators in NWP models. We explore different ML methods and DNN architectures for classification and regression on data gathered from an implementation of the KMP scheme from an idealized simulation of a cloud model.

In Section 2, the dataset is described in detail, including basic statistics, interpretation of its variables, and an overview of *no-operation* samples. In Section 3, the methodology is described with a justification for implementing a classification model, a description of how the data was prepared, and a description of the DNN models. Section 4 conveys the results gathered with a comparison of the DNN and baseline models for both the classification and regression tasks. Finally, in Section 5 we discuss the meaning of our results, possible future work on the subject, and the limitations of our study.

II. DATASET

The dataset consists of 24 million input and output samples from the KMP scheme from a three-dimensional cloud model. The data is made up of seven input variables and 4 target variables. Table 1 displays basic metrics for each variable. The correlation matrix of the variables is shown in figure 1. The physical interpretation for each variable is:

- potential temperature (theta): The temperature a parcel of air at a given pressure would attain if adiabatically brought to a reference pressure of 1000 hPa.
- qc: The dry mixing ratio of cloud liquid
- qv: The dry mixing ratio of water vapor
- qr: The dry mixing ratio of rain liquid
- rho dry: The density of dry air at the given height
- height: Height measured in meters
- Exner pressure: A dimensionless pressure quantity

VARIABLE	UNITS	MEAN	STANDARD DEVIATION	MINIMUM VALUE	MAXIMUM VALUE
theta in	K	353.75	52.71	295.06	483.10
qc in	mol/mol	5.90e-06	7.22e-05	0.0	0.0038
qv in	mol/mol	0.0024	0.0042	7.83e-06	0.016
qr in	mol/mol	2.51e-05	0.00041	0.0	0.021
rho dry	kg/m^3	0.48	0.30	0.093	1.13
Height	m	10000.0	5771.70	250.0	19750.0
Exner pressure	N/A	0.70	0.17	0.44	0.99
theta out	K	353.76	52.71	294.94	483.10
qc out	mol/mol	5.86e-06	7.28e-05	0.0	0.0038
qv out	mol/mol	0.0024	0.0042	7.83e-06	0.016
qr out	mol/mol	2.49e-05	0.00041	0.0	0.021

Table 1: List of all variables, units, mean values, standard deviation, maximum values, and minimum values.

The high correlation between the input and target variables theta, qc, qv, qr suggests that the KMP scheme performs a simple, linear operation on the input variables. Further, numerical analysis of the data reveals that a majority of samples are 'no operation' (no-ops) samples. We found that approximately 93% of the 24 million samples have equal values of input and target variables. That is, 93% of the samples satisfy:

$$MAX(\,|T_{in} - T_{out}|,\ |qc_{in} - qc_{out}|,\ |qv_{in} - qv_{out}|,\ |qr_{in} - qr_{out}|\,) < 1e-16.$$

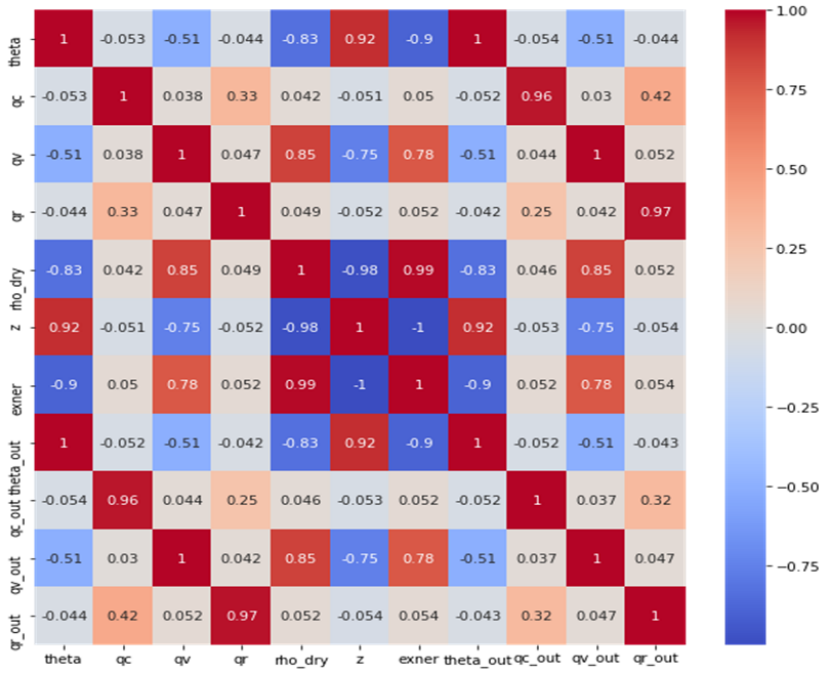

FIGURE 1: The correlation matrix of all variables. Notice the high correlation between the inputs and targets for temperature, qc, qv, and qr in the top right of the matrix.

The miniscule change occurring in these no-operation samples is displayed by computing the absolute difference for each variable over the no-operation samples:

$$|\Delta V| = \sum_{no-operation\ samples} (|x_i - y_i|)$$

where V is a given variable, x_i is the input, and y_i is the target. Doing so for the mixing ratios qc, qv, and qr gives:

$$|\Delta qc_{no-op}| = 2.974747786606549e-13$$
$$|\Delta qv_{no-op}| = 5.351217651503384e-13$$
$$|\Delta qr_{no-op}| = 1.736223377611681e-13$$

This is negligible change over 93% of the data and it would be unreasonable to expect a machine learning model to capture this behavior when performing regression. In comparison, the total

change between inputs and targets of the operation samples is given by computing the absolute difference for each variable:

$$|\Delta V| = \sum_{operation\ samples} (|x_i - y_i|)$$

Doing so over qc, qv, and qr, gives:

$$|\Delta qc_{op}| = 26.309639413255496$$

$$|\Delta qv_{op}| = 60.63209451999513$$

$$|\Delta qr_{op}| = 121.72817254607185$$

III. METHODS

Preparing the data for the ML models was straightforward. First, every sample was assigned a boolean value to identify it as an operation or no operation sample based on the condition in section 2. Second, the data was scaled using min-max normalization which fits the data to be within a [0,1] scale. Third, we removed samples from the dataset to create a desired ratio between operation and no operation samples. The specific ratios used depended on the task to be performed on the data. Lastly, the data was shuffled and split into training and testing sets with a test size 0f 30%. We explored classification models to predict the boolean value assigned to each sample and regression models to predict the output variables.

A deep neural network is commonly defined to be a neural network with many hidden layers. Here, we use two DNN architectures for both classification and regression. The first is a network with ten 64-node hidden layers(10x64) and the other is a network with six 100-node hidden layers(6x100). By today's standards, these DNNs are small. However, given the simplicity of the KMP scheme, these DNNs should have more than enough complexity to capture the behavior of the data.

The discrepancy in the number of operation and no-operation samples created the need for a ML method to classify operation and no-operation samples. For a KMP emulator to be of use, a classifier must be implemented to identify operation samples based on the input variables. To this end, we tested the two DNNs described above and five baseline classification models: logistic regression, stochastic gradient descent (SGD) classifier, support vector classifier (SVC), decision tree classifier (DTC), and a perceptron with a single 100 node hidden layer. To gauge how the model accuracy is affected by the ratio of operation to no-operation samples, three sets of classifiers were trained and tested. One set with a dataset with an operation to no-operation ratio of 1:1, another set with a ratio of 1:3, and a third with a ratio of 1:10. To create these datasets, every operation sample (1702636 samples) was mixed with a random sampling of no-operation samples into a single dataset with the desired ratio. This dataset was then shuffled and split into a training and testing set with a testing set size of 30%. The classification models were then trained and tested on these three datasets to obtain three sets of classifiers. The models were trained to

identify operation samples, i.e., a 'True' prediction corresponds to an operation sample and a 'False' prediction to a no-operation sample. Comparison of these models was done by computing the accuracy and confusion matrices for each model on their respective testing dataset.

The procedures used to obtain the regression model results are quite similar to those used for the classification models. Three baseline models were tested: Linear Regressor (LR), decision tree regressor (DTR), and a perceptron with a single 100 node layer. Again, we tested the 10x64 and 6x100 DNN architectures for regression. We also explored the effect of the ratio of operation to no-operation samples for regression. Using the same method to mix operation and no-operation samples, four datasets were created with ratios of 0 (100% operation samples), 10:1, 3:1, and 1:1. The five models were trained and tested on each dataset, creating four sets of regressors. Comparison of the regression models is done by computing both the averaged and variable mean absolute error (MAE) of the models testing set predictions.

IV. RESULTS

A. Classifiers

The testing set accuracy scores for the classification models is plotted versus the percent of no-operation samples in Figure 2. Clearly, an increase in the number of no-operation samples in the dataset correlated to an increase in testing accuracy. However, this result is not surprising since identifying no-operation samples is a rather trivial task given enough data. The classifier with the highest accuracies for every dataset was the DTC. The two DNNs produced almost identical accuracy scores for every dataset and outperformed every model excluding the DTC. Since there are two classes, random choice could achieve 50% accuracy average. Every model exceeds this threshold and performs better than random labeling.

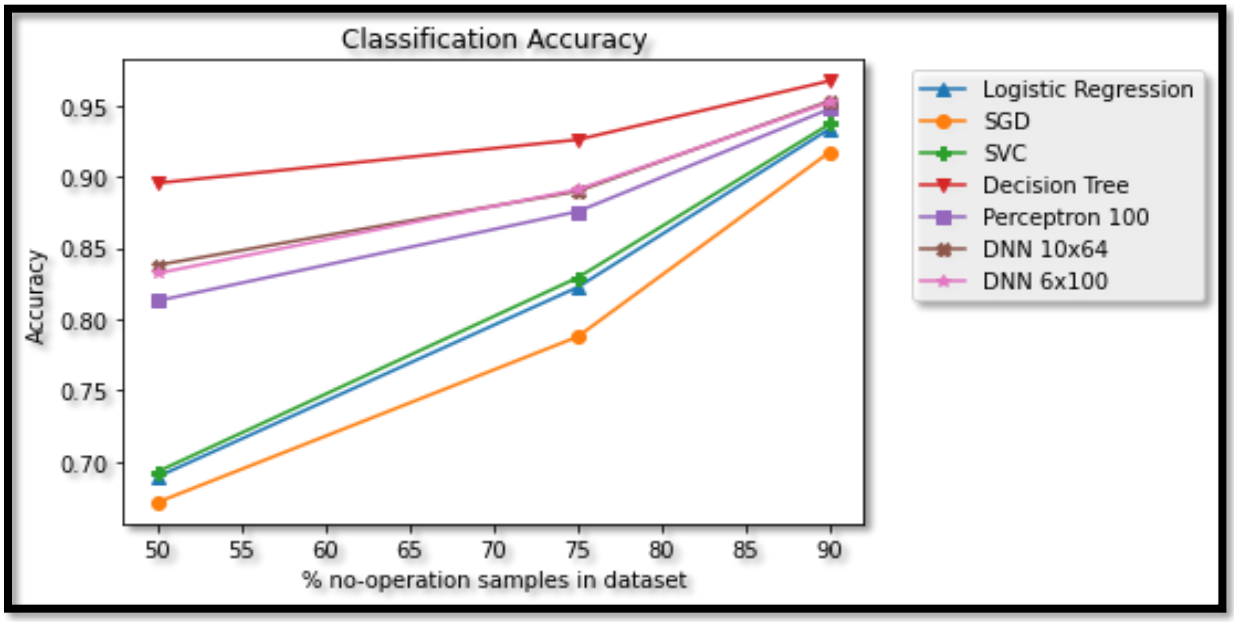

FIGURE 2: The percent of no-operation samples in the dataset vs. testing set accuracy scores for classification models.

Figure 3 offers a comparison of the confusion matrices of the DTC and DNNs. The DTC achieves a higher accuracy compared to the DNNs by reducing the number of both false-positive and false-negative predictions. Across all three models there is a sharp decrease in the number of false-positive results as the no-operation percentage increases. There is also a sizable decrease in the number of false-negatives predicted as the no-operation percentage increases.

If this KMP emulator was to be implemented within a NWP model, a highly accurate classification would act as a filter to the regression model. If the classifier predicts the inputs correspond with performing an operation, they are passed on to the regression model to produce outputs. If the classifier determines the inputs are no-operation data, then the KMP scheme is bypassed. In this way, the classifier determines whether the emulator should compute a result or not. A false-positive occurs when a no-operation sample is predicted to be an operation sample. In this case, the no-operation sample is passed to the regressor, and an operation is performed when it should not have. This may give rise to error if the regressor has not seen many no-operation samples in training. A false-negative prediction by the classifier means an operation was incorrectly labeled. In this case, the KMP will be bypassed when an operation should have occurred. Obviously, this can lead to error in the NWP model since the KMP variable were not updated correctly. For the reasons above, it is vital to have a highly accurate classification model. The results encourage using a high percent of no-operation data to achieve the best classification accuracy.

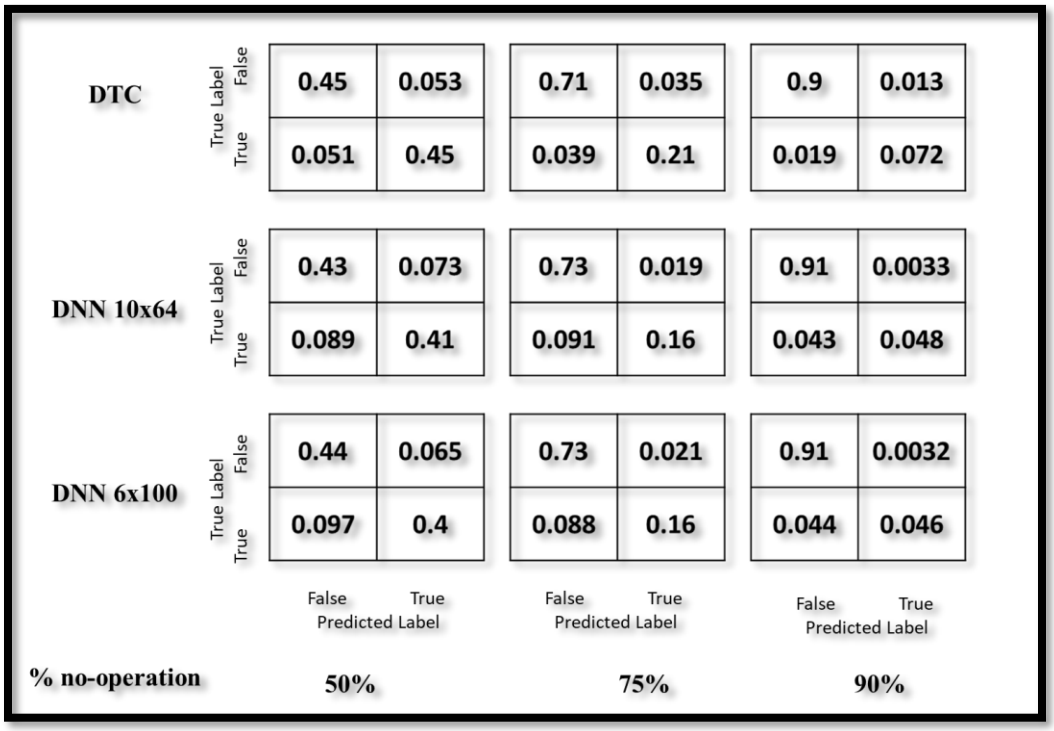

FIGURE 3: Confusion matrices for the decision tree classifier and the two deep neural networks for the three testing datasets.

B. Regressors

Figure 4 displays the mean absolute errors of the regression models for the four sets of data. In general, as the percentage of no-operation samples increases, the MAE decreases. The DTR outperformed the other models, including the DNNs. The 6x100 DNN produced lower MAE values compared to the 10x64 DNN for three datasets, excluding the 33% no-operation dataset. The linear regression model produced significantly larger MAE scores. Table 2 displays a comparison of the MAE values for each target variable for the DTR and DNN models trained and tested with 10% no-operation data.

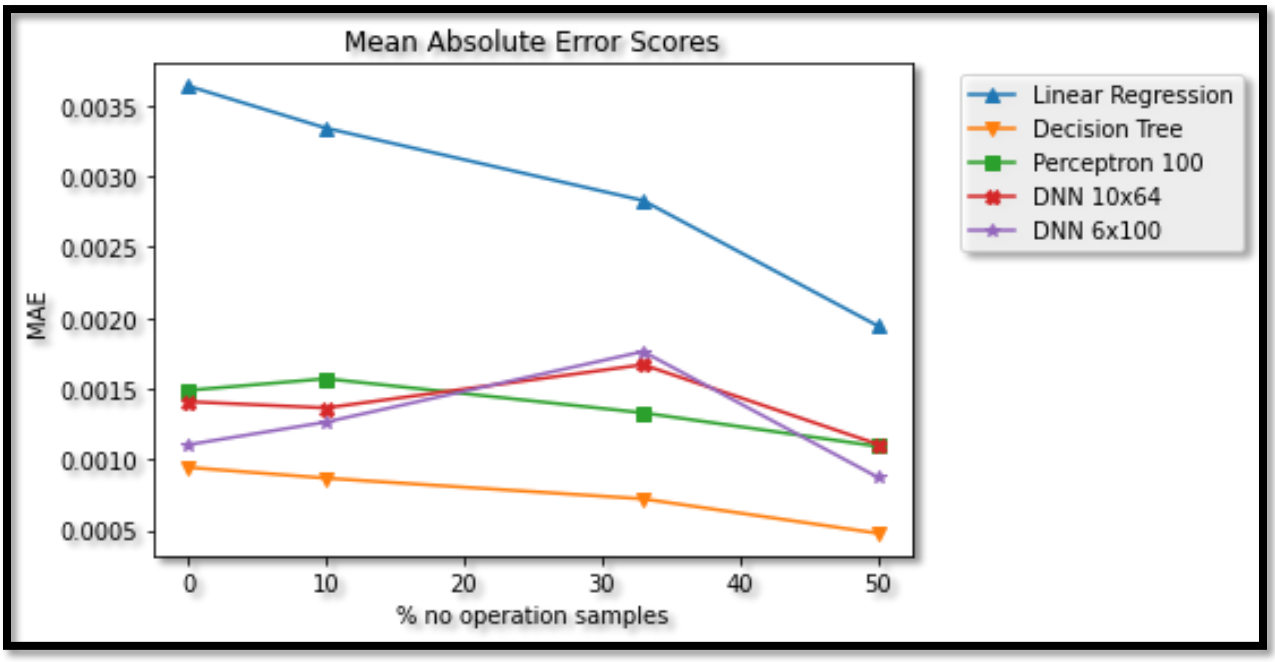

FIGURE 4: The percentage of no-operation samples in the dataset vs. mean absolute error for each model

The values displayed in Table 2 that the models produced lower MAE scores for theta and qv than qc and qr. The table also illustrates that the DTR produces the lowest MAE values for every target variable except qc, where the 10x64 DNN performs marginally better. The results suggest that a large number of no-operation samples leads to better MAE test scores. However, there are reasons to be skeptical of such a conclusion. First, it is possible that a large number of no-operation samples can contaminate a machine learning model to overfit data or fit to incorrect patterns within the data. Second, in general, it is best practice to train a regression model on data which best represents that which it will encounter upon implementation. For our purposes that means it would be best practice to train the regression models on a training set with a low number of no-operation samples. To be exact, we should train the regression models on a dataset with a no-operation percentage equal to the rate of false-positives predicted by the classification model. As explained above, an implementation of the KMP emulator would consist of a classifier and regression model. The false positives produced by the classifier would be the only no-operation samples a regressor would encounter upon implementation. Using this methodology suggests that the regression

models should be trained and optimized on datasets with a low percentage (1-6%) of no-operation samples.

MODEL	THETA MAE	QC MAE	QV MAE	QR MAE
DTR	0.00037234	0.00108047	0.00062389	0.00139221
10 X 64 DNN	0.00101882	0.00104978	0.00107548	0.00230854
6 X 100 DNN	0.00082258	0.00131478	0.00068855	0.0022279

Table 2: The mean absolute error for each target variable produced by the decision tree regressor and DNN regressors trained on a 10% no-operation dataset.

V. CONCLUSIONS

In this paper we evaluated various method to implement an emulator is for the KMP scheme. The data set consisted of 24 million samples gathered from a cloud model which implements the KMP scheme. The samples contained an overwhelming majority of 'no-operation' samples where the inputs do not undergo a transformation, satisfying:

$$MAX(\,|T_{in} - T_{out}|,\ |qc_{in} - qc_{out}|,\ |qv_{in} - qv_{out}|,\ |qr_{in} - qr_{out}|\,) < 1e-16.$$

These no-operation samples can contribute to precision noise contamination and overfitting in regression models. As such, the emulator is made up of two ML models: a classifier and a regressor. The classifier was trained to identify samples which undergo a transformation, or so called 'operation' samples. Two DNN classifiers were trained: a ten layer network with 64 nodes per layer and a six layer network with 100 nodes per layer. Five baseline classifiers were also implemented: a logistic regression model, a stochastic gradient classifier, a support vector classifier, a decision tree classifier, and a perceptron with a single hidden layer of 100 nodes. The models were trained on three different datasets with a varying number of no-operation samples. For classification, the DNNs produced similar accuracy to one another, and higher accuracy scores compared to every model excluding the decision tree classifier. We found the decision tree classifier produced the highest accuracy and a larger number of no-operation samples correlated to improved test accuracy scores for all models. For regression, the DNNs and single layer perceptron produced similar mean absolute error scores. Again, the decision tree regressor produced the lowest mean absolute error scores and a larger number of no-operation samples correlated to lower error values in testing. However, we discourage the use of a large number of no-operation samples as it may contribute to overfitting and does not comply with data science's best practices. The work described above only provides a glimpse at the possibilities of ML emulators and demonstrates some of the challenges researchers in the climate, data, and computational sciences may encounter when implementing them. Namely, the quality of the data gathered can pose a significant hurdle to creating effective emulators. Clearly, this research is of limited scope and much work is left to be done in this particular subject. Future research could take many avenues including implementing ML emulators for other NWP model schemes, testing other DNN architectures, or developing best practices for gathering and preparing data from these schemes. Considering how well the decision tree models performed for both tasks, it is likely worthwhile to explore the applicability of decision tree-based emulators. We hope the work

presented here motivates and informs those in the computational and climate sciences to explore the possibilities of ML emulators.

REFERENCES

1. P. Arpaia, G. Azzopardi, F. Blanc, G. Bregliozzi, X. Buffat, L. Coyle, E. Fol, F. Giordano, M. Giovannozzi, T. Pieloni, R. Prevete, S. Redaelli, B. Salvachua, B. Salvant, M. Schenk, M. Solfaroli Camillocci, R. Tomás, G. Valentino, F.F. Van der Veken, J. Wenninger. 'Machine learning for beam dynamics studies at the CERN Large Hadron Collider'. *Nuclear Instruments and Methods in Physics Research Section A: Accelerators, Spectrometers, Detectors and Associated Equipment*, Volume 985, 2021. https://doi.org/10.1016/j.nima.2020.164652.

2. A.A. Masrur Ahmed, Ravinesh C. Deo, Qi Feng, Afshin Ghahramani, Nawin Raj, Zhenliang Yin, Linshan Yang. 'Deep learning hybrid model with Boruta-Random forest optimiser algorithm for streamflow forecasting with climate mode indices, rainfall, and periodicity'. *Journal of Hydrology*, Volume 599, 2021. https://doi.org/10.1016/j.jhydrol.2021.126350.

3. S. Rasp, P.D. Dueben, S. Scher, J. A. Weyn, S. Mouatadid, N. Thuerey. 'WeatherBench: A Benchmark Data Set for Data-Driven Weather Forecasting'. *Journal of Advances in Modeling Earth Systems*, Volume 12, 2020. http://dx.doi.org/10.1029/2020MS002203.

4. C. K. Sønderby, L. Espeholt, J. Heek, M. Dehghani, A. Oliver, T. Salimans, S. Agrawal and J. Hickey and N. Kalchbrenner. 'MetNet: A Neural Weather Model for Precipitation Forecasting'. 2020. https://arxiv.org/abs/2003.12140.

5. Jarno Mielikainen, Bormin Huang, Jun Wang, H.-L. Allen Huang, Mitchell D. Goldberg. 'Compute unified device architecture (CUDA)-based parallelization of WRF Kessler cloud microphysics scheme'. *Computers & Geosciences,* Volume 52, 2013. https://doi.org/10.1016/j.cageo.2012.10.006.

ACKNOWLEDGEMENT

This research used resources of the Oak Ridge Leadership Computing Facility, which is a DOE Office of Science User Facility, and the resources of the Compute and Data Environment for Science (CADES) at the Oak Ridge National Laboratory, both supported by the Office of Science of the U.S. Department of Energy under Contract DE-AC05-00OR22725.