Geosci. Model Dev. Discuss.,
https://doi.org/10.5194/gmd-2019-79-RC1, 2019
© Author(s) 2019. This work is distributed under
the Creative Commons Attribution 4.0 License.

Geoscientific Model Development Discussions

Interactive comment on "Fast domain-aware neural network emulation of a planetary boundary layer parameterization in a numerical weather forecast model" *by* Jiali Wang et al.

Anonymous Referee #1

Received and published: 5 June 2019

This paper presents a series of neural networks designed to reproduce the output of the YSU PBL parameterization in the WRF model. The goal is to use the NN as a proxy model to reduce the computational cost of running the WRF model. The premise of the work is interesting and worthy of publication. There are several points of clarification and confusion however.

1. The topology of the models is confusing. The FFN is fine, but the size of the hidden layers should be noted. The two hierarchical models appear to be a series of nested single layer neural networks that output at each level. The goal is to enforce layer specificity, though I am not sure why this cannot also happen in the FFN as with what I

assume are a larger number of internal weights and fully connected, it should be able to encode this information as well.

2. I don't understand why the FFN performance is so much worse. If the intermediate layers are sized large enough, then it should have a much larger number of connections and be able to encode more than the hierarchical models. It appears to cut out in training much earlier however. Is this just overfitting due to larger number of connections vs training data? If the amount of data was vastly increased, would we expect FFN to eventually overtake the performance of the hierarchical models?

3. The writeup of the evaluation is a bit confusing. In particular L231-236. I assume it means that you trained using a single grid location, then applied the model to multiple grid points within 800km. If so this should be made more clear. Also why specify individual sites. One could calculate performance on all grid points within 800km. While doing this, it would be useful to see the drop off in performance as a function of distance. The last two plots start down this path, but with the density of points in a model, it should be straight forward to give performance as a function of distance from training point within the 800km range.

4. Can the authors comment on where they see this being put in an online model? It seems like the round trip to and from a GPU (IO) would cause a much bigger delay than just calculating the YSU parameterization in place. Offline this is not as much of a concern as all the data can be preloaded, but when there is a round trip at every model time step, it seems like the IO would become the predominant factor, and not the computation.

5. I assume this is meant to be used as a proxy model for YSU when you are interested in fiddling with a different portion of the model and just want something "good enough" that is computationally cheap (For instance, you're examining microphysical parameterizations, and don't care about PBL explicitly). Is there a concern that the feedback loops with being off by as much as this NN is (up to 60% for some parame-

ters, though much less for most) would cause the output from YSU and this model to diverge quite quickly when being run as a replacement for YSU? If so is this just meant to be a parallel option for a parameterization, or as a drop in for YSU?

Interactive comment on Geosci. Model Dev. Discuss., https://doi.org/10.5194/gmd-2019-79, 2019.