Geosci. Model Dev. Discuss.,
https://doi.org/10.5194/gmd-2019-327-AC1, 2020

Geoscientific Model Development Discussions

Interactive comment on "RadNet 1.0: Exploring deep learning architectures for longwave radiative transfer" *by* Ying Liu et al.

Ying Liu et al.

ying.liu.sweden@gmail.com

Received and published: 14 April 2020

article [utf8]inputenc enumitem color url

Reply to the Reviewer 1 comments for: RadNet: Exploring deep learning architectures for longwave radiative transfer

We thank the reviewer for their comments.

1 General Comments

1. `Study setup You have chosen to use ERA profiles as input which of course gives you realistic profiles. However, this comes with two major limitations: a) You have to manually create out-of-sample profiles (Dataset 2) which are potentially unrealistic (see comment below). b) You are limited to single-column model experiments for coupled verification. While I don't expect you to do this for this revision, I feel like you have the chance for a more insightful experimental setup using CliMT. You could have setup a global CliMT simulation that is somewhat realistic and used the RRTMG output from that simulation as training data. Then you could have also ran the same experiment`

with increased SSTs for a more realistic climate change experiment. Lastly, you could have used your RadNet in directly in the global simulation for a 3D coupled experiment. I think the results from such an experiment could be really insightful for the ML parameterization community.

Response: We thank the reviewer for these suggestions. We would like to clarify that:

a) Our validation dataset was not specifically created with a climate change scenario in mind. Rather, our aim was to explore the response of NNs when perturbations are added to the input profiles. Radiative heating profiles can be fairly noisy since optical parameters can change quite drastically in the vertical, and these changes need not be highly correlated. For example, the presence/absence of clouds, hydrometeors, aerosols, horizontal advection of water vapour can cause localised perturbations which are not strongly correlated vertically. That being said, we agree that our validation dataset represents an extreme example of this behaviour. We have added a paragraph in Section 2.2 to explain the reasoning that went into creating the perturbed dataset.

b) There is nothing in our setup which limits our simulations to a single column. This is because radiative transfer is assumed to have no contribution from adjacent columns (as we have described in Lines 91–92.). Our motivation to use a single column model for validation has been described in the manuscript in line 350: it allows us to study the evolution and possible errors in greater detail.

We thank the reviewer for their suggestions regarding the proposed experimental setup, which are natural extension to the work presented here, and we look forward to following this up in future work.

Lines Changed: 129–133

2. Training data estimation for line-by-line computation As you mentioned in the paper, ideally one would use line-by-line radiation computations to train from. Since line-by-line is really expensive you would be limited in the amount of training data you can generate. Would it be possible to make an estimate of what a realistic amount of line-by-line data would be and whether this amount of data would be enough to train your networks

   Response: The dataset used to train the network in this paper is around 12 million training samples and 3.5 million validation samples. Thus, the total dataset size generated is around 15 million samples. RRTMG requires 0.37 ms to generate a sample as per our profiling results (Table 1). The total time to generate all samples amounts to around 1.5 hours. The performance of a line-by-line radiative transfer code is dependent on the spectral resolution it uses; However, it is typically around 2-3 orders of magnitude slower than correlated-k codes (see Table 2 in https://doi.org/10.3390/rs11090994). Using a 10-core machine (for example) to generate the samples, we would require it to run continuously for around 2 weeks to generate the required samples. This estimate does not take into consideration any memory-related constraints that may arise.

3. Neural network architectures One of the key goals of this paper is to compare different NN architectures, particularly fully connected versus convolutional. You present 5+ different architectures. I have several questions about these:

   (a) Model E is the same as Model C but with zero-padding but why do the number of channels also increase in the table from 4 to 6? I assume this is an error since the number of parameters also stays the same.

Response: The channel number is correct. The increase is caused by the zero-padding. However, it is good that you have pointed out the parameter numbers. We made an error in calculating the numbers. Here are the correct number of parameters with calculation steps. The number of parameters are also supported by the speedup presented in the later section.

Model A: $240 * 512 + 512 * 1024 + 1024 * 512 + 512 * 60 = 1202176$

Model B: $240*512+512*1024+1024*2048+2048*1024+1024*512+512*60 = 5396480$

Model C: $3 * 3 * 128 + 3 * 3 * 128 * 256 + 56 * 1 * 256 * 512 + 512 * 60 = 7666816$

Model D: $3*3*128+3*3*128*256+3*3*256*256+54*1*256*512+512*60 = 7994496$

Model E: $3 * 3 * 128 + 3 * 3 * 128 * 256 + 58 * 2 * 256 * 512 + 512 * 60 = 15531136$ We have also made the appropriate changes in the manuscript.

Lines Changed: Table 1, Row 5

(b) You could have tried a fully convolutional network by using zero-padding on each layer. That way you don’t need the final fully connected layer, which means a lot fewer parameters. I would be interested to see how such a network would perform.

Response: We do not think that adding padding to all conv layers will act the same as fully connected layers. The way to construct a fully connected layer using conv layer is to use conv filters the same size as the input layer with stride 1. So that each conv filter computes the input only once covering all vectors then the number of conv filters aggregates, e.g. 128. Then, it will act like 128 neuron fully connected layer.

(c) However, one problem I see with CNNs in this context is that they are based on translational invariance i.e. the computations are the same along the entirety of the vector. But with non-uniform grids like the one you are using is there any reason to assume that translation invariance should hold?

Response: We are unsure what the reviewer means here. We presume they are referring to the fact that the pressure grid is non-uniform, and therefore translation along the pressure axis is not an invariant.

Our understanding is that translational invariance is a *consequence* of the architecture of CNNs. This is because of the architecture of convolutional layers which carry out the same computation along the entire vector. Then, in the later fully connected layers, features extracted from convolutional layers are aggregated with respect to their location in the input vector using different weights. Finally, the prediction of radiation is produced.

Translation invariance is a useful property to have when the aim is to identify features regardless of their location. However, our focus was on the ability of CNNs to capture local features well rather than their ability to produce translation invariant (or equivariant) representations of the input. We believe that this is a useful feature to have given that radiative heating fields can have sharp, local features due to the presence of clouds and other radiatively active substances.

(d) Another architecture you could try is a Resnet using skip connections.

Response: We would expect that Resnet will give more accuracy in this use case. However, we expect that such a model will be even slower than the model E given that Resnet-50 has over 25 million parameters. Therefore, it is unlikely to speed up radiative transfer calculations.

(e) You mention you use PReLUs to avoid vanishing gradients. Did you observe vanishing gradients for normal ReLUs? That would surprise me since your networks are not very deep.

Response: The reviewer is correct that we did not observe vanishing gradients in all models. We only observed an indication of vanishing gradients in our deepest model. In addition, PReLUs facilitate faster and more stable convergence. To make the comparison fair, we have chosen the same activation function in all models.

(f) `Did you observe overfitting (train MSE < valid MSE)? Your validation dataset is from 2015–2016 while your train dataset is from 1979–1985, which means that the base state is probably changing. This will make it hard to check for overfitting since even with a model that does not overfit one would expect the train loss to be lower. A better test of overfitting would be to take a random subset of the train dataset. The reason I am interested in this is that I am surprised a bigger network (B vs A) did not give you a better score. If you are not yet overfitting, bigger networks should always result in better losses, or am I missing something? You mention overfitting when talking about the generalization experiments but that still doesn't explain why the skill for Dataset 1 wouldn't be better for model B.`

Response: We agree with the reviewer that it is harder to explicitly check for over-fitting using different periods of data using standard definitions. However, we believe our approach is a more practical check of the models' ability to generalize since we would like the model to perform well in different climate states.

2 Specific Comments

- `Line 56: You could also mention this recent paper: https://arxiv.org/abs/2001.03151. Also this follow up paper highlights some interesting issues: https://onlinelibrary.wiley.com/doi/abs/10.1029/2019MS001711`

  Response: We thank the reviewer for these references. They have been added to our bibliography.

  Lines Changed: 55–57

- `Line 90: Typo "columm"`

  Response: Corrected.

  Lines Changed: 91

- `Line 102: I think it would be useful at this point to mention the horizontal and vertical resolution of ERA.`

  Response: We have added the horizontal and vertical resolution.

  Lines Changed: 101–103

- `Line 105: So the size of your training dataset is around 12 million? Please mention this explicitly?`

  Response: We thank the reviewer for pointing this out. Yes, our training dataset is 12 million samples. We have added number in the manuscript.

  Lines Changed: 109

- `Line 110: If I understand correctly there is no vertical correlation in your perturbation, right? Are the resulting profiles therefore "unrealistic"? How do you think this` `affects the network's generalization? Do you think that your results are representative of the sorts of perturbations you would experience in real climate change (increased T)?`

Response: We agree that our perturbed dataset is an extreme case in that it assumes zero correlation between perturbations at different levels. However, as mentioned previously, changed in the optical properties of the atmosphere can be fairly "noisy" with low correlation in the vertical. In that sense, we believe our perturbed dataset is an extreme case of a regularly occurring phenomenon. Our choice of zero correlation aims at providing a conservative estimate: if the neural networks can perform adequately in these conditions, they will likely perform as well or better when there perturbations are correlated and have fewer effective degrees of freedom. Nonetheless, we agree that these perturbed profiles affect the network's ability to generalize, and produces issues that we have noted (Line 328 in the original manuscript). In a climate change scenario, given that our ability to simulate the vertical and horizontal distribution of clouds is still poor, a milder version of perturbations we have used might be relevant.

- `Line 150: Actually, convolutions are a form of regularization by introducing an architectural constraint. In fact, a convolution layer is simply a subset of a fully connected layer with shared weights (https://medium.com/impactai/cnns-from-different-viewpoints-fab7f52d159c). But yeah, in effect you end up focusing on local features.`

  Response: We thank the reviewer for this insight. It is certainly will help us going forward to think of CNNs in this manner.

- `Table 1: I think it would be helpful to list all the Models`

`(including F) in the table.`

  Response: Model F is now in the table.

  Lines Changed: Table 1, row 6

- `Data: Great that you included an example file in the repo. But maybe you could also add instructions on how to download the full dataset you used.`

  Response: We have added the URL for the ERA Interim data in the Code Availability section.

- `Code: Kudos for including code. I looked at the implementation of the networks. Maybe just a recommendation: I think the code could have been a lot easier and clearer if you used Keras instead of plain TF. Especially with TF 2 its now very easy to take all the precoded parts of Keras and implement your own layers, etc. using TF. All the manually coded layers in your code (matmul, etc.) make it very hard to follow when a simple Dense() would suffice. This would make the code easier to read for others.`

  Response: The skeleton of keras code illustrating the NN structure for easier reading is added in the repository. It is in the model.py file line 243-255. But it is recommended to use the original implementation since it is the base for the results presented in this paper.