Supporting Information for "Detection of forced change within combined climate fields using explainable neural networks"

Jamin K. Rader[1], Elizabeth A. Barnes[1], Imme Ebert-Uphoff[2,3], Chuck Anderson[4]

[1]Department of Atmospheric Science, Colorado State University, Fort Collins, CO, USA

[2]Cooperative Institute for Research in the Atmosphere, Colorado State University, Fort Collins, CO, USA

[3]Department of Electrical and Computer Engineering, Colorado State University, Fort Collins, CO, USA

[4]Department of Computer Science, Colorado State University, Fort Collins, CO, USA

Contents of this file

Corresponding author: Jamin K. Rader, (jamin.rader@colostate.edu)

April 5, 2022, 7:46pm

S1 Neural Network Specifications

All units of the neural networks use a rectified linear unit (ReLU) activation function, except for the output layer which uses a soft-max layer to rescale the final outputs of the neural network such that they sum to one. We train the neural networks using the binary cross-entropy loss between the predicted class likelihoods and the correct class membership weights, such that the loss function is minimized when the two are equal. More information on the ReLU activation function, the soft-max layer, and the loss function can be found in sections A1, A2, and A3 of Barnes et al. (2020), respectively.

The neural networks were trained using the Keras Adam optimizer, an adaptive stochastic gradient descent algorithm (Kingma & Ba, 2014). We used a learning rate that started at 0.001 and decayed linearly to 0.0005 over the span of 150 epochs. Although the Adam optimizer is designed to alter the learning rate based on the momentum of training, the decaying learning rate allowed the neural networks to train more quickly with improved performance. Weights and biases were initialized using random values from a normal distribution.

As discussed in Section 3.1, our neural networks are fully connected with two hidden layers and 10 nodes each in each layer. We found that this architecture allowed the neural networks to capture forced change better than a linear model or even simpler architectures, such as neural networks with only one hidden layer or five nodes in each layer (Figure S2). The additional performance offered by more complicated architectures was small and increased the computational resources needed for training. We elected to stick with the simplest model that performed well with minimal computational expense. These neural networks can be trained on standard laptop

or desktop computers in two to ten minutes depending on the input field, making them extremely accessible to those in the climate science community.

As discussed in Section 3.2 and Figure S4, we applied a ridge penalty (L2 regularization) to the input layer (see Barnes et al., 2020). The ridge penalty was selected such that the time of emergence detected by the neural networks was the earliest. All input vectors used a ridge penalty of 0.1, except for seasonal-mean temperature and precipitation combined input vector, for which the TOE was earlier for a ridge penalty of 0.01 (see Figure S4).

Summary of Neural Network Specifications

Number of Hidden Layers	2
Number of Nodes in Each Hidden Layer	10
Hidden Layer Activation Function	ReLU (Rectified Linear Unit)
Output Layer Activation Function	Softmax
Ridge Penalty (applied to the weights of the first hidden layer)	0.01 for seasonal-mean temperature and precip 0.1 for all other input fields
Loss Function	Binary Cross-entropy
Optimizer	Adam, tensorflow.keras.optimizers.Adam
Learning Rate	Started==s at 0.001, decaying linearly to 0.0005
Number of Epochs	150

S2 Selection of Neural Network Hyperparameters

We explored a range of values for several neural network hyperparameters such as the learning rate (from 10^-4 to 10^-1), the number of epochs (up to 1000), the ridge penalty (from 0 to 1, see Figure S4), and the neural network architecture, where we examined the performance of neural networks with 1, 2, or 3 hidden layers, and 5, 10, 20 or 50 nodes in each hidden layer (see Figure S2). To choose these hyperparameters we employed a strategy similar to leave-p-out cross-validation which is commonly used in the atmospheric sciences (Celisse & Robin, 2008). Specifically, we used 10 different train/test splits to explore the hyperparameter space

and optimize the performance of our neural networks. Using 10 different train/test splits, rather than just one, ensures that our hyperparameter selections are not overfitting to any one specific way the climate models can be split into training and testing sets. Once the best hyperparameter choices were made, we then used another 100 train/test splits for the results of this study, all of which differed from the train/test splits used for tuning.

S3 K-means Clustering

Before applying k-means clustering, all LRP maps are converted into binary maps. Every grid point on each LRP map is assigned a one or a zero depending on whether its relevance value is greater than or less than the mean relevance across all maps and grid points. In this way, ones indicate regions of high relevance, and zeros indicate regions of low relevance. K-means clustering is then applied to these binary LRP maps (3200 in total, samples from 32 climate models for 100 neural networks). We used Sci-Kit Learn's sklearn.cluster.KMeans function (version 0.22.1) in Python with 100 different initializations and all other choices were left as default (Pedregosa et al., 2011). The results for K = 2 are shown in the main paper. Using K = 2 identified two clusters that were near-equal in size, and several runs of k-means with different random initial conditions yielded near-identical results. Clustering for K = 3, 4, 5, 6, 7, 8, and 32 was also explored, however the results for three or more clusters were less physically consistent.

S4 Additional Observational Datasets

In addition to the observational datasets in Section 2.2, we also test two additional precipitation observations in Figures S5 and S6. First, we use the European Center for Medium-Range Weather Forecasts' ERA5 global reanalysis (Hersbach et al., 2020) at 6-hour resolution to con-

struct observational monthly mean precipitation fields from 1980 to the present. Second, we use the Japan Meteorological Agency's Japanese 55-year Reanalysis (JRA55; Kobayashi et al., 2015) mean 3-hour precipitation forecasts to construct observational monthly mean precipitation fields from 1959 to the present.

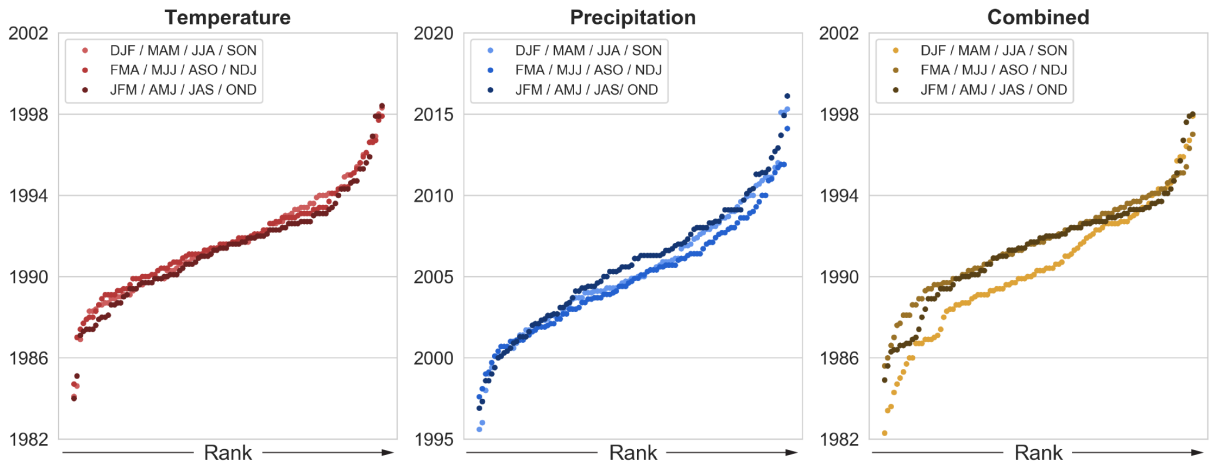

Figure S1. TOE detected by the neural networks given different definitions of season. As in Figure 5d-f, but for each possible three-month combination of seasons. All three definitions lead to similar TOE when neural networks are trained on global maps of temperature or precipitation. When temperature and precipitation are combined, meteorological seasons lead to the earliest detection of forced change.

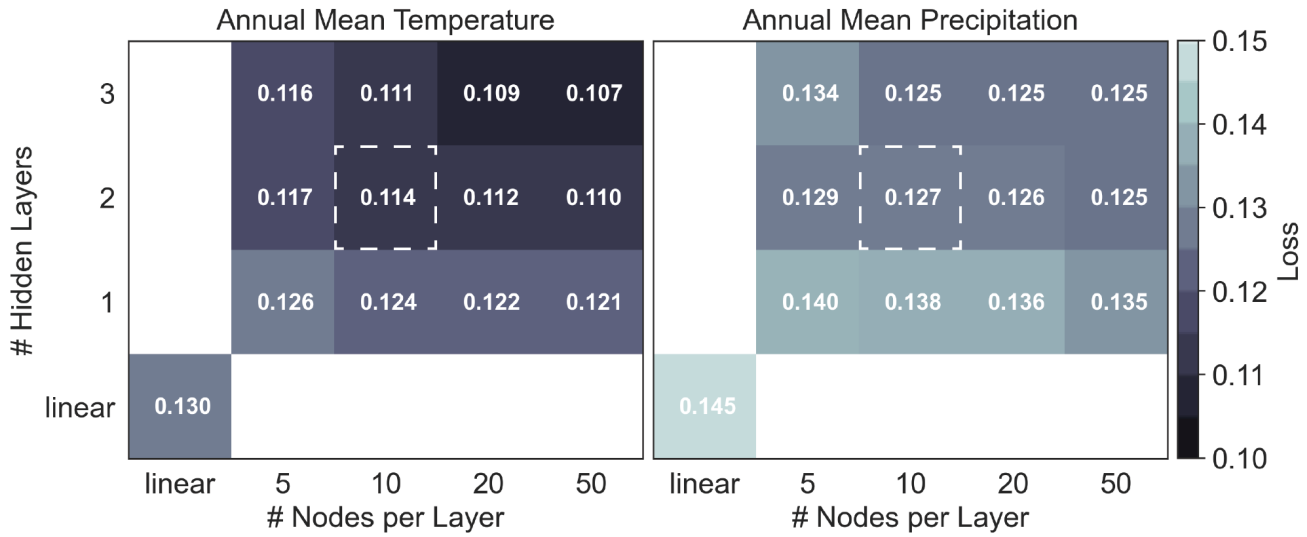

Figure S2. Skill across various neural network architectures. The mean testing binary cross-entropy loss for ten trained neural networks with different train/test splits for 12 different neural network architectures and one linear model for annual-mean temperature and annual-mean precipitation. The white box indicates the neural network architecture that was used in the main text (two hidden layers, 10 nodes each).

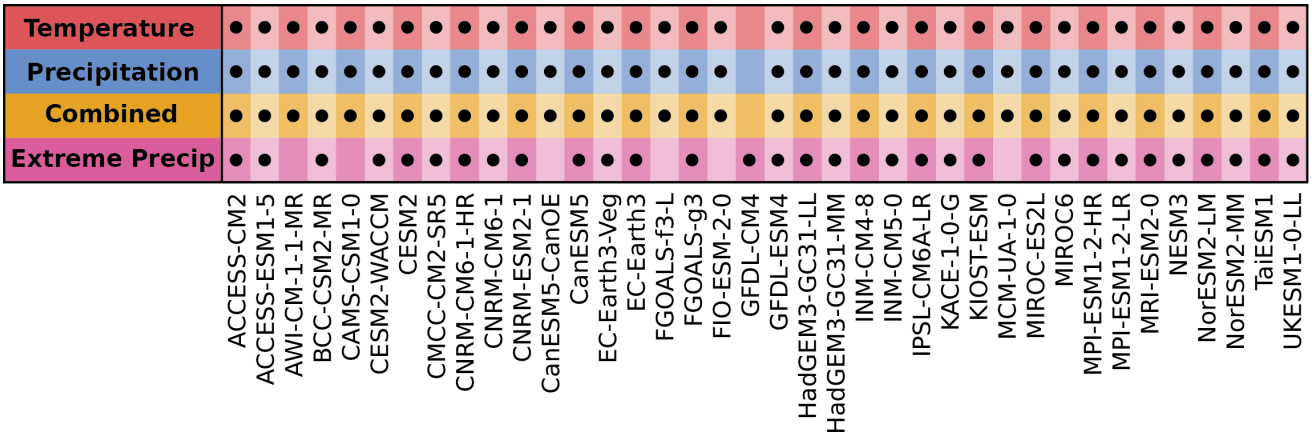

Figure S3. Climate models used for each input variable. Temperature, precipitation, and temperature and precipitation combined used the same 37 CMIP6 climate models. Extreme precipitation fields came from 32 climate models for which daily precipitation fields were available.

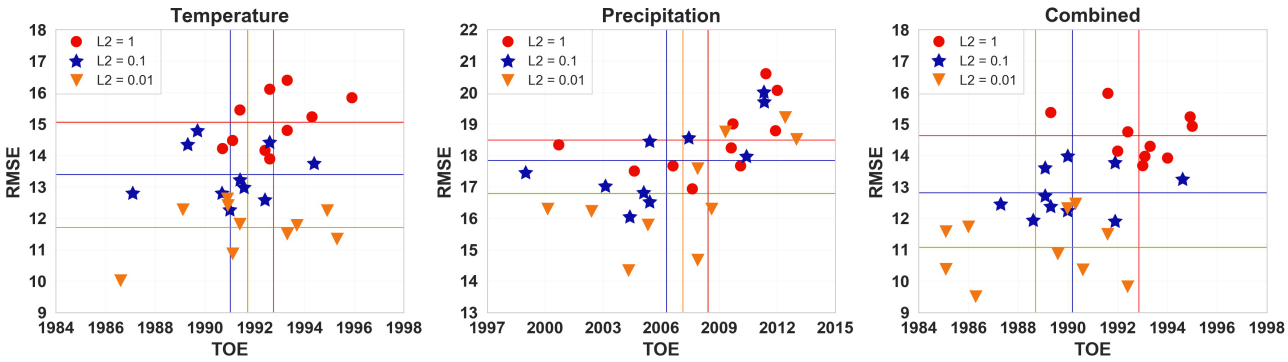

Figure S4. TOE and RMSE for various ridge penalties. The sensitivity of RMSE and TOE to the ridge (L2) penalty used for 10 neural networks trained on seasonal-mean maps of (a) temperature, (b) precipitation, and (c) temperature and precipitation combined. Each plot shows the RMSE and TOE for neural networks trained with a ridge penalty of 1, 0.1, and 0.01 (denoted by red circles, blue stars, and orange triangles, respectively). The mean RMSE and TOE for all 10 neural networks are indicated by the horizontal and vertical lines. Each neural network for a given variable/ridge penalty differs only in which climate models were part of the training and testing sets. While a ridge penalty of 0.01 leads to the smallest mean RMSE in all cases, using a higher ridge penalty of 0.1 leads to earlier detection of forced change for temperature and precipitation input vectors. As a result, we choose to use the ridge penalties corresponding to an earlier TOE.

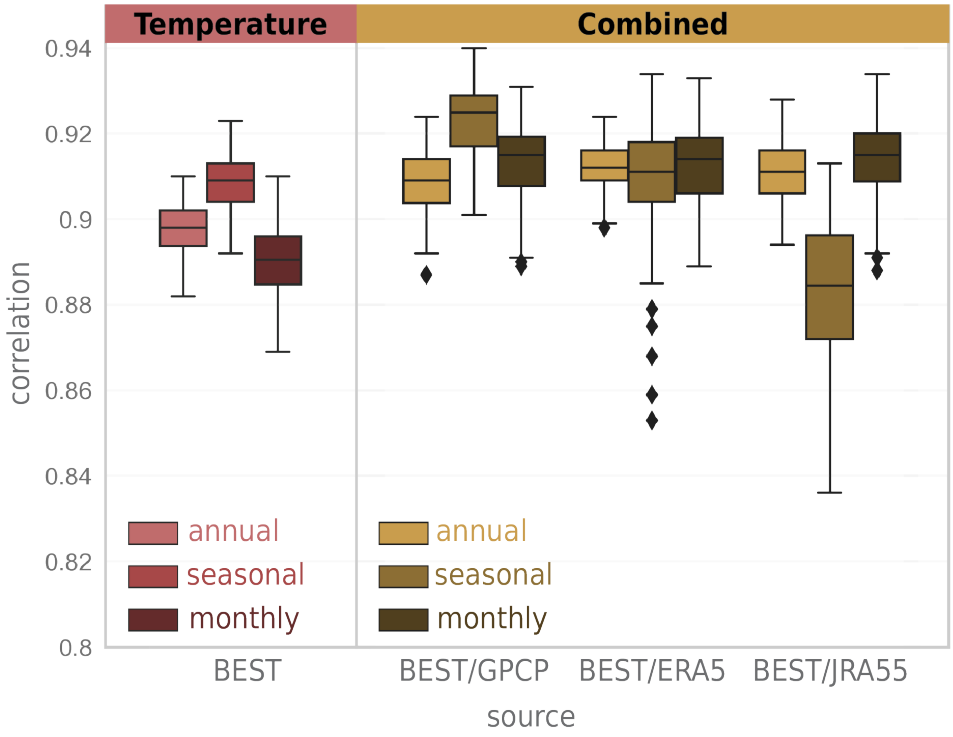

Figure S5. Sensitivity of observational correlations to the source of precipitation observations: temperature and precipitation combined. Pearson correlations of the actual years with the years predicted by 100 trained neural networks given observations of temperature and precipitation. Correlations were computed for all years beginning in 1980 where observational data exists for all variables. The box plots indicate the first, second, and third quartile statistics, and the whiskers denote 1.5 times the interquartile range, or the minumum/maximum value, whichever is less extreme. The observational correlations for seasonal-mean combined neural networks are sensitive to the dataset of choice, as observational correlations are higher for GPCP than ERA5 or JRA55. This is not the case for the annual-mean and monthly-mean combined neural networks, which have approximately the same correlations regardless of the source of the observations. This is because the seasonal-mean combined neural networks rely on precipitation to predict the year, while the annual-mean and monthly-mean combined neural networks do not, as shown in Figure 5.

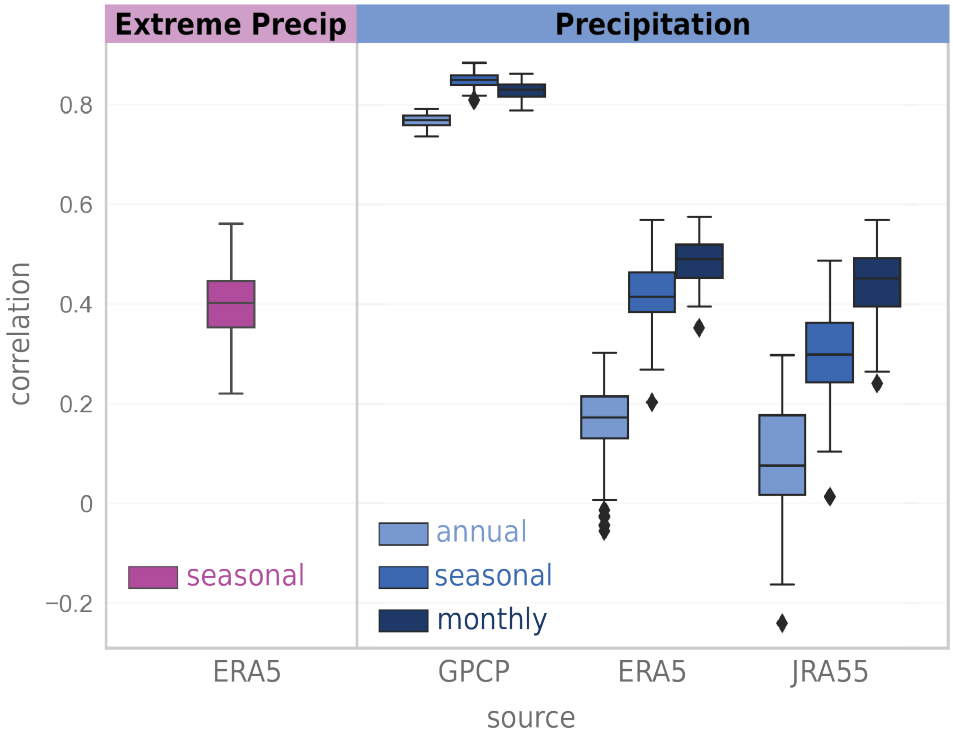

Figure S6. Sensitivity of observational correlations to the source of precipitation observations: precipitation only. Pearson correlations of the actual years with the years predicted by 100 trained neural networks given observations of precipitation. Correlations were computed for all years beginning in 1980 where observational data exists for all variables. The box plots indicate the first, second, and third quartile statistics, and the whiskers denote 1.5 times the interquartile range, or the minumum/maximum value, whichever is less extreme. The observational correlations are sensitive to the source of precipitation data. Correlations are highest for GPCP, followed by ERA5 and JRA55. The observational correlations for ERA5 seasonal-mean extreme precipitation are similar to those for ERA5 seasonal-mean precipitation.

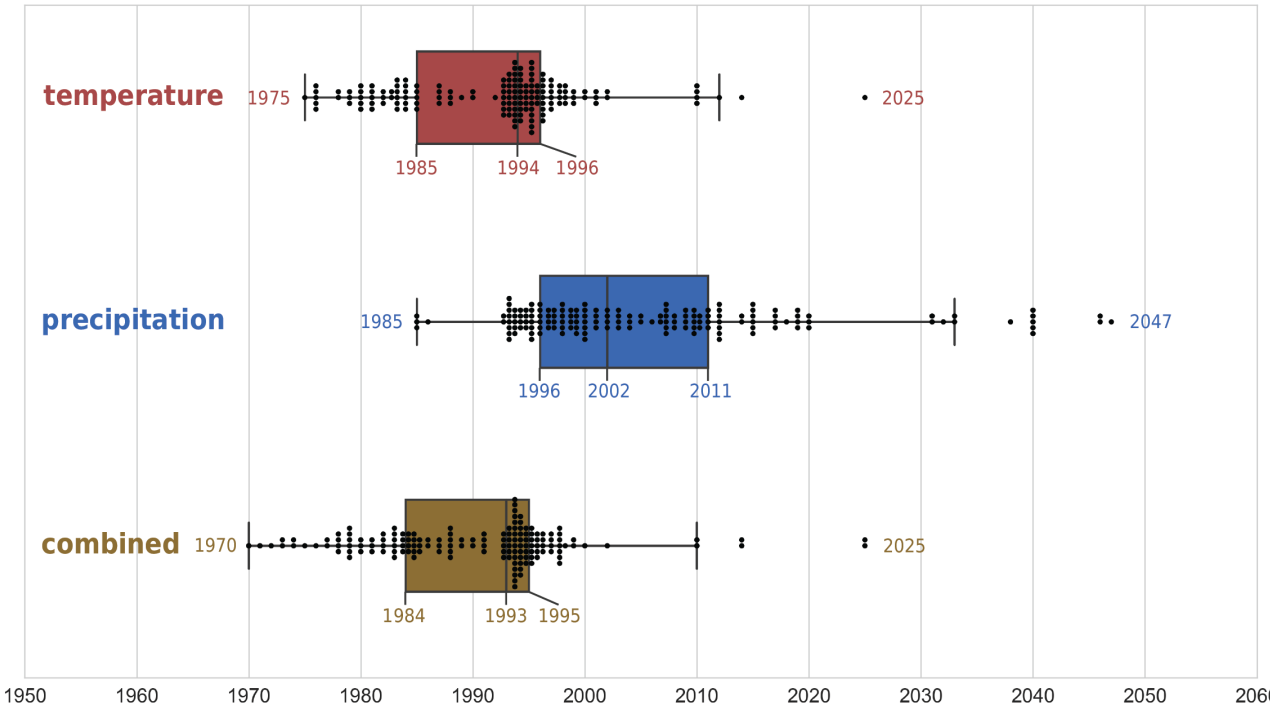

Figure S7. Time of emergence for seasonal-mean fields. TOE was calculated for each climate model in the testing sets of 100 trained neural networks. Each dot represents five (rounded up) occurrences of the associated TOE year (i.e. one dot represents 1-5 occurrences, two dots represent 6-10 occurrences, and so on). For added clarity, box plots indicate the first, second, and third quartiles of the TOEs for each model, and whiskers denote 1.5 times the interquartile range, or the minimum/maximum point, whichever is less extreme.

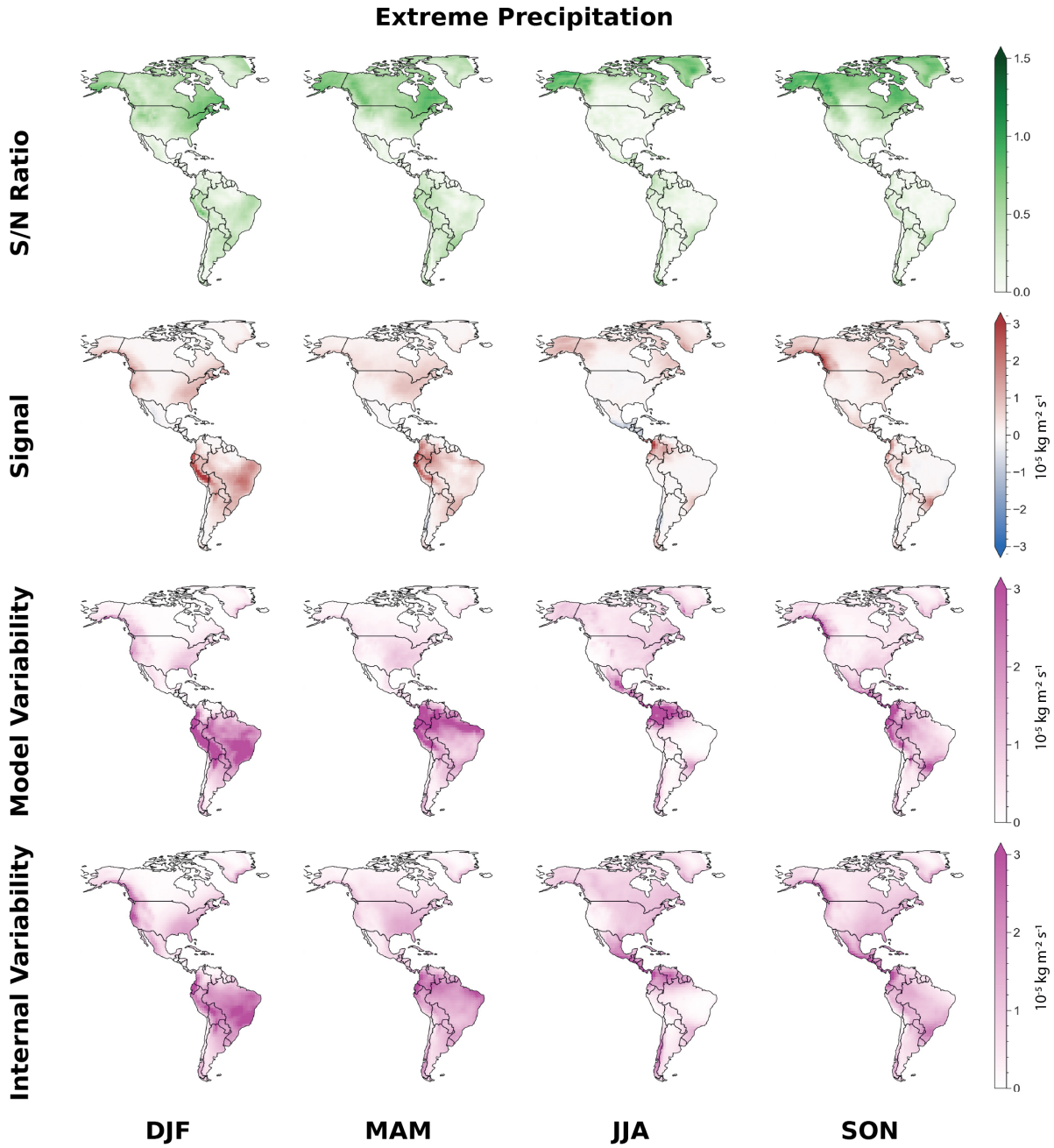

Figure S8. Signal and noise for extreme precipitation over the Americas. Plots of S/N ratio, and its components (signal, climate model variability, and internal variability) for extreme precipitation in each season over North and South America. The signal is most clear over the northern-most latitudes. The S/N ratio is below 1.5 in all seasons indicating that there is considerable noise relative to the signal of change.

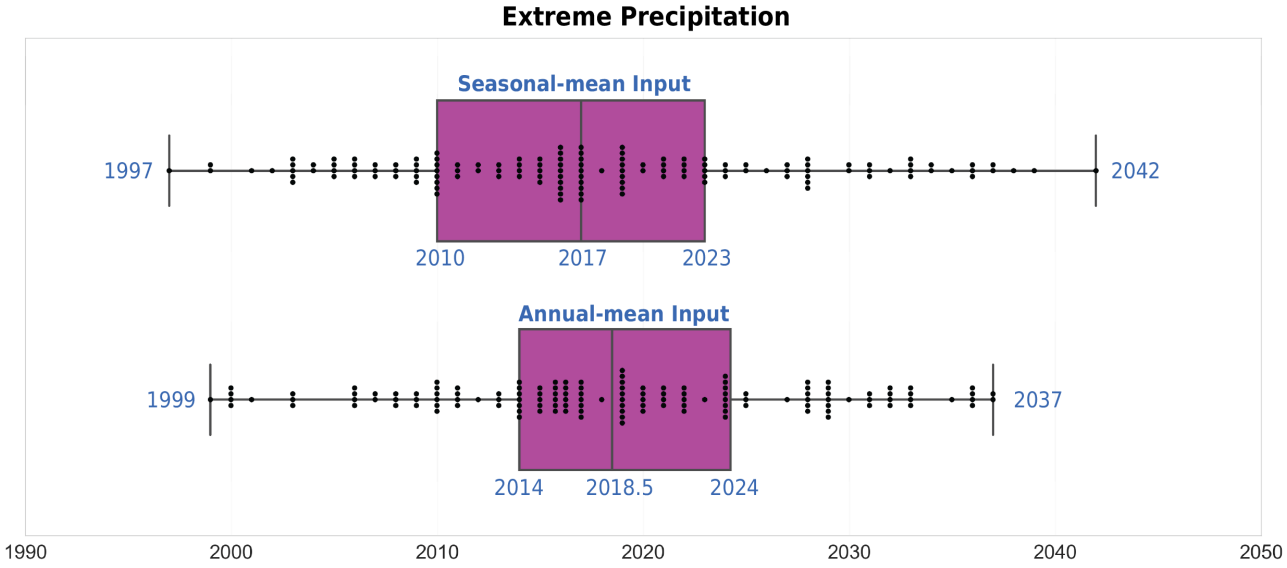

Figure S9. Time of emergence for extreme precipitation over the Americas. TOE was calculated for each climate model in the testing sets of 100 trained neural networks. Each dot represents five (rounded up) occurrences of the associated TOE year (i.e. one dot represents 1-5 occurrences, two dots represent 6-10 occurrences, and so on). For added clarity, box plots indicate the first, second, and third quartiles of the TOEs for each model, and whiskers denote 1.5 times the interquartile range, or the minimum/maximum point, whichever is less extreme.

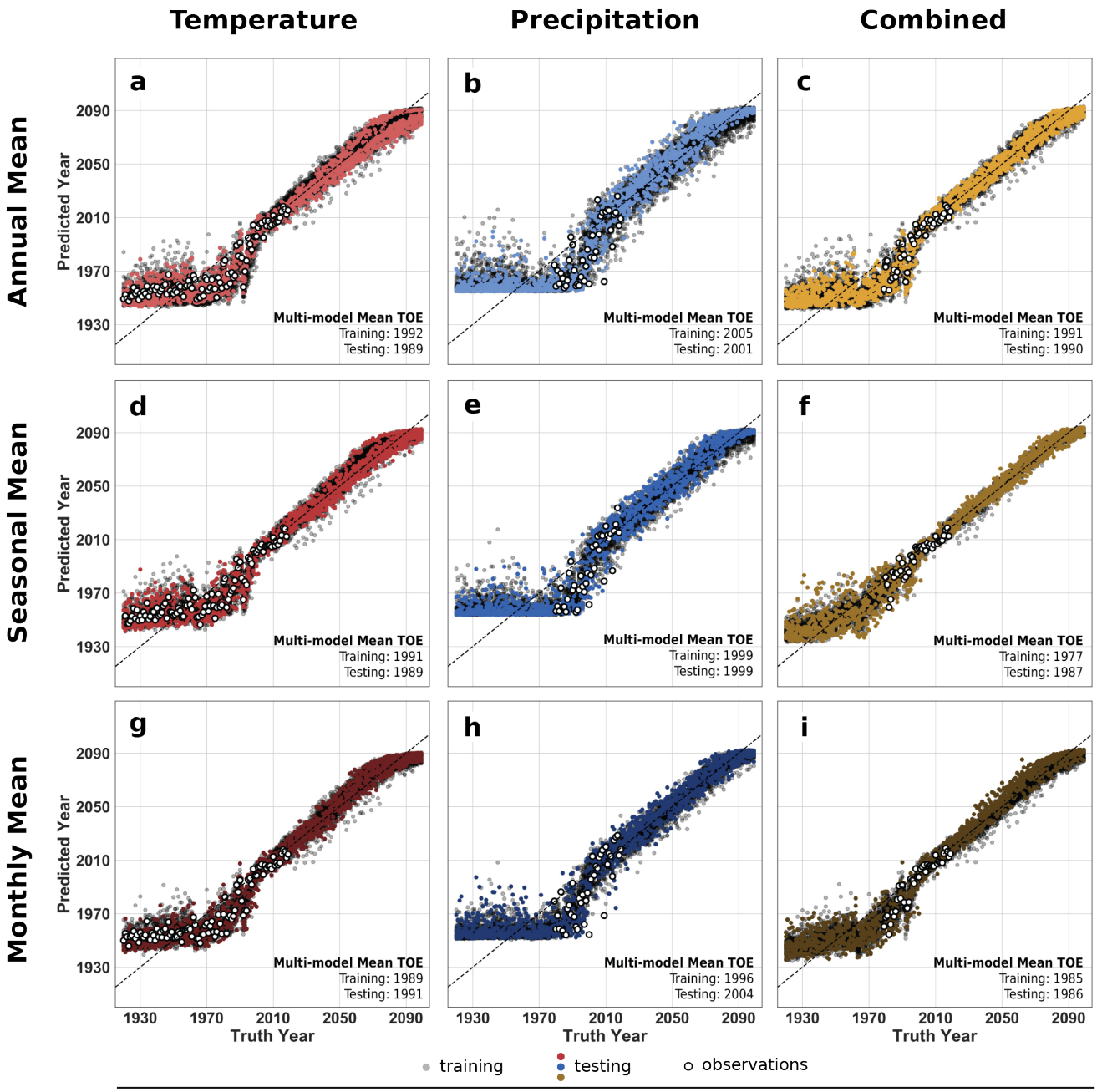

Figure S10. Neural network output for temperature and precipitation (with training data included). Same as Figure 3 with training data included. Year predicted by the neural network (y-axis) versus the truth year (x-axis) for temperature (a, d, g), precipitation (b, e, h), and temperature and precipitation combined (c, f, i). Input maps include annual-mean data (a, b, c), seasonal-mean data (d, e, f), and monthly-mean data (g, h, i). Training data is shown in gray, testing data is shown in color, and observations are shown in white.

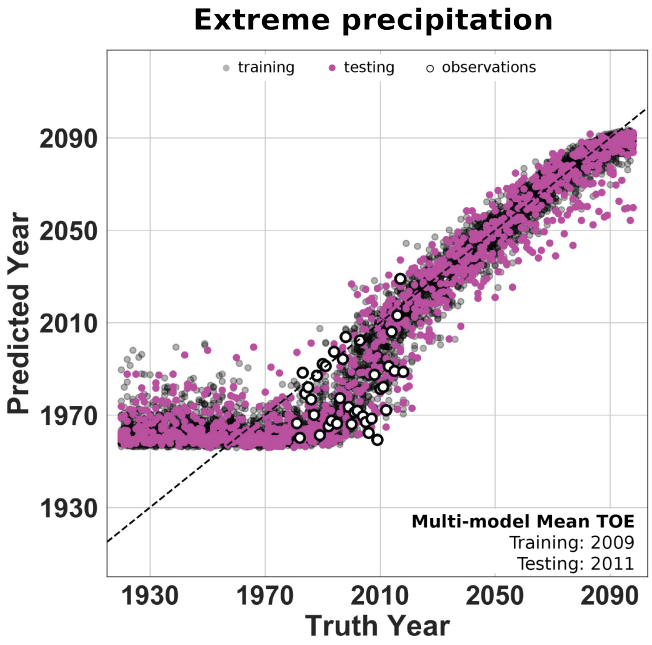

Figure S11. Neural network output for extreme precipitation (with training data included). Same as Figure 8 with training data included. Year predicted by the neural network (y-axis) versus the truth year (x-axis) given seasonal-mean maps of extreme precipitation. Training data is shown in gray, testing data is shown in pink, and observations are shown in white.

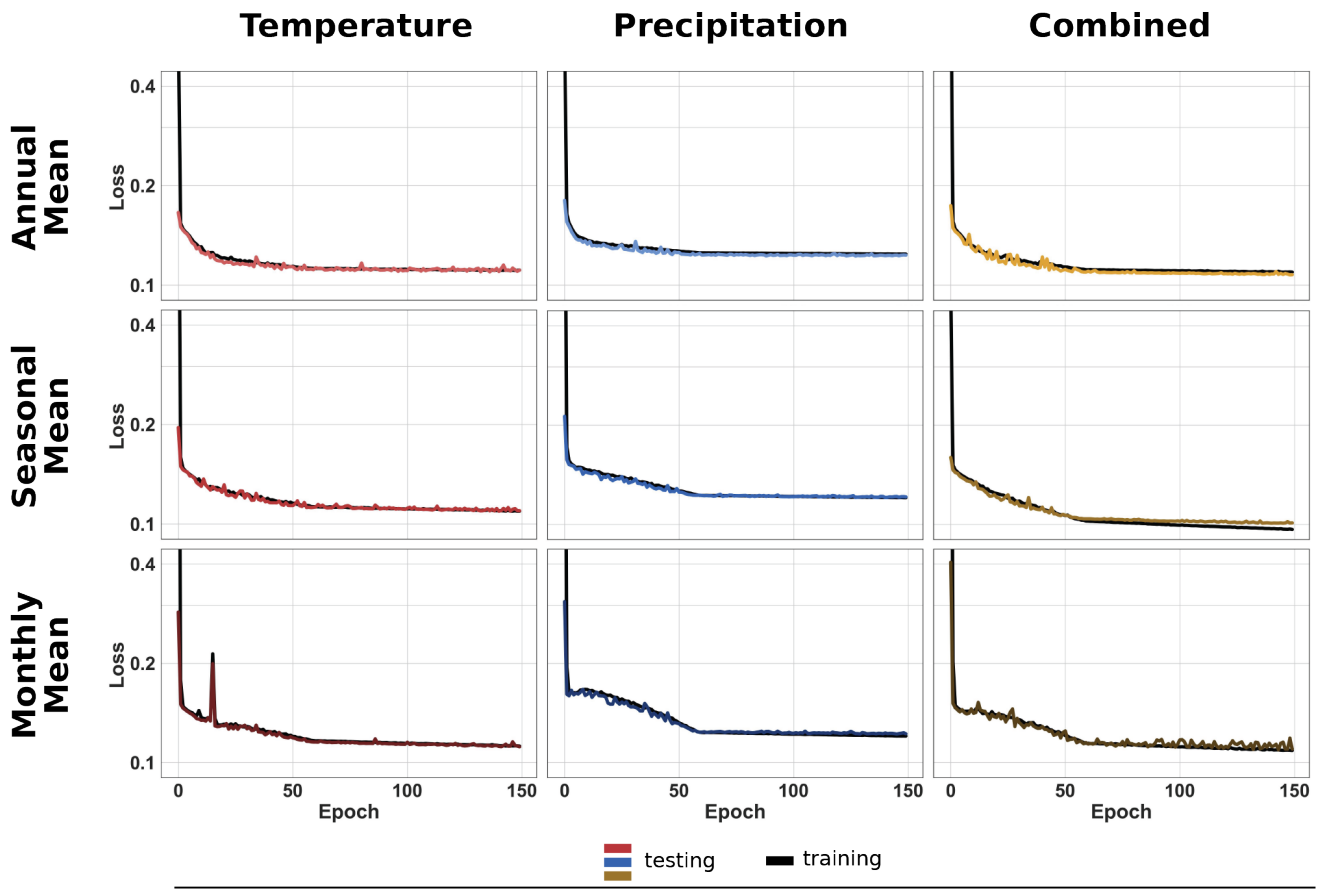

Figure S12. Learning curves for temperature and precipitation. Binary cross-entropy loss versus epoch of training for the training and testing data for the nine trained neural networks shown in Figure 3, Figure S10.

References

Barnes, E. A., Toms, B., Hurrell, J. W., Ebert-Uphoff, I., Anderson, C., & Anderson, D. (2020, September). Indicator patterns of forced change learned by an artificial neural network. *J. Adv. Model. Earth Syst.*, *12*(9), e2020MS002195. doi: 10.1029/2020ms002195

Celisse, A., & Robin, S. (2008, January). Nonparametric density estimation by exact leave-p-out cross-validation. *Comput. Stat. Data Anal.*, *52*(5), 2350–2368. doi: 10.1016/j.csda.2007.10.002

Hersbach, H., Bell, B., Berrisford, P., Hirahara, S., Horányi, A., Muñoz-Sabater, J., ... Jean-Noël Thépaut (2020, July). The ERA5 global reanalysis. *Quart. J. Roy. Meteor. Soc.*, *146*(730), 1999–2049. doi: 10.1002/qj.3803

Kingma, D. P., & Ba, J. (2014, December). Adam: A method for stochastic optimization.

Kobayashi, S., Ota, Y., Harada, Y., Ebita, A., Moriya, M., Onoda, H., ... Takahashi, K. (2015). The JRA-55 reanalysis: General specifications and basic characteristics. . *2*, *93*(1), 5–48. doi: 10.2151/jmsj.2015-001

Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., ... Duchesnay, E. (2011). Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research*, *12*, 2825–2830.