Supporting Information for "Interpretable Deep Learning for Probabilistic MJO Prediction"

Antoine Delaunay[1]and Hannah M. Christensen[2]

[1]Department of Applied Mathematics, Ecole Polytechnique, Palaiseau, France

[2]Department of Physics, University of Oxford, Oxford, UK

Contents of this file

1. Text S1 to S6
2. Figures S1 to S12
3. Tables S1 to S4

Introduction

In this Supporting Information we provide further details on the methodology used in our study. Text S1 and Table S1 provide details of the Subseasonal-to-seasonal (S2S) prediction project data used as a benchmark for the CNN performance. Text S2 describes the observational data and preprocessing used to train the model. Text S3 provides more details of the CNN forecasting model, focusing on the techniques used to represent epistemic and aleatoric uncertainty, and Figure S2 shows the CNN architecture. Figure S3 shows sensitivity of the CNN performance to chosen input fields. Text S4 details the

PatternNet algorithm. Text S5 details the validation metrics. Text S6 proves the validity of the DropBlock approach in place of standard dropout for convolutional layers.

We also provide further results to support our conclusions. Figure S4 shows the bivariate correlation skill for the CNN and S2S models. Figure S1 shows the phase diagram corresponding to the decaying and propagating events analysed in Section 3.2 of the manuscript, while Figures S5–S7 show further interpretation of the CNN forecasts for those events. Figures S8–S12 show further results concerning predictors of uncertainty in MJO forecasts for outgoing longwave radiation (OLR), 850 hPa geopotential (Z850), and sea surface temperature (SST).

Text S1. *Subseasonal-to-Seasonal forecast model data*

We select four representative models from the Subseasonal-to-Seasonal (S2S) prediction project database (F. Vitart et al., 2017) for comparison with the CNN. The database consists of near real-time operational ensemble forecasts and reforecasts from 11 centres. As the operational models are continuously improved, the skill of the forecasts evolves in time. For that reason, we select the reforecasts for comparison with the CNN. Reforecasts are forecasts made retrospectively using a single up-to-date version of the dynamical model.

Details of the available reforecast data for the selected models are presented in Table S1. Some observations have to be made: first, all models do not have the same number of members, so to be consistent we decided to restrict the number of members to 10. Second, as the computational cost is heavy, the reforecasts are not made every day and consequently each model has a different reforecasting period and time range. This is an issue which is difficult to overcome but with a large enough number of days, we should still be able to make fair comparisons.

Text S2. *Observational data and preprocessing*

We train the CNN using atmospheric data from the ECMWF ERA5-Reanalysis dataset (H. Hersbach et al., 2018a), (H. Hersbach et al., 2018b). The inputs are maps of daily averaged fields from 1979 to 2019 with a spatial coverage of 0 – 360°E, 20°S – 20°N on a 2.5° x 2.5° grid. We only use ERA5 data over the satellite era for which accurate estimates of OLR are available. Selected variables are: zonal wind at 200 hPa and 850 hPa (UA200, UA850), Outgoing Long-Wave Radiation (OLR), Sea Surface Temperature

(SST), Specific Humidity at 400 hPa (SHUM400), Geopotential at 850 hPa (Z850), and Downward Long-Wave Radiation at the surface (DLR). For UA200, UA850 and OLR, we apply the RMM preprocessing transform of (Wheeler & Hendon, 2004) to leave only subseasonal anomalies: the time mean, the first three Fourier harmonics and the 120-day running mean are removed sequentially. For SST, we subtract the climatological mean (for each date of the calendar year, we compute the average over the same date for all the years in the training dataset), and set all inland grid points to zero. The raw data is used for SHUM400, Z850, and DLR, allowing the network to learn seasonal variations in MJO predictability. Finally for every variable, we rescale the inputs to between 0 and 1 independently at every gridpoint with a Min-Max scaling to ensure the stability of the training.

Our choice of input fields for the CNN was guided by an iterative procedure. The first network we trained took as input the three variables used to define the RMM index: UA200, UA850 and OLR subseasonal anomalies. Subsequent networks were trained using one or more additional variables, and the predictive performance of the network assessed. Supporting Figure S3 shows the relative benefit of including each of the additional input variables selected for the final network: sea surface temperature anomalies (SST), daily downwelling long-wave radiative forcing (DLR), daily geopotential at 850 hPa (Z850), and specific humidity at 400 hPa (SHUM400). We compared the performance of the final network to a network trained on DLR, Z850 and SHUM400 *anomalies* instead of *means*, but found this degraded performance. We also considered including the values of fields

at earlier timesteps (5, 10 days before), but found this did not improve the network's performance, and instead led to overfitting.

Text S3. *The CNN forecasting model*

For an initial date t and a forecast range τ, let $\mathbf{x}_t$ be the input at the date t, and $\mathbf{y}_{t+\tau}$ be the observed RMM indices, $\mathbf{y}_{t+\tau} = (\mathrm{RMM1}_{t+\tau}, \mathrm{RMM2}_{t+\tau})$ at the chosen lead time, τ. The input $\mathbf{x}_t$ is a series of gridded maps representing physical quantities (variables) for each date t as a function of latitude and longitude. We train a separate network for each forecast range τ, where τ takes discrete values: $\tau = 1, 3, 5, 10, 15, 20, 25, 30, 35$ days.

Aleatoric uncertainty is caused by the chaotic nature of the system. Physically, we recognise that the input variables supplied to the CNN are a subset of all possible variables, and only include information on scales larger than the resolution of the input maps, such that the future state of the MJO is not a deterministic function of these inputs[1]. This uncertainty is a property of the data and thus irreducible, regardless of the model's training. It is also heteroscedastic, or state-dependent. The predicted aleatoric uncertainty is included as an output of the CNN. We assume the RMM indices follow a Gaussian bivariate distribution with a null correlation between RMM1 and RMM2 (Wheeler & Hendon, 2004). The probabilistic network therefore has a 4-neuron output consisting of the forecast mean, $\boldsymbol{\mu}_{t+\tau}$, and variance $\boldsymbol{\sigma}^2_{a\ t+\tau}$, where the first and second entries of $\boldsymbol{\mu}$ and $\boldsymbol{\sigma}^2_a$ correspond to RMM1 and RMM2 respectively. Aleatoric uncertainty is accounted for in the loss function: the model is trained by maximising the log-likelihood:

$$L = \frac{1}{N}\sum_{t=1}^{N} -\frac{1}{2}[ln(|\mathbf{\Sigma}_{t+\tau}|) + (\mathbf{y}_{t+\tau} - \boldsymbol{\mu}_{t+\tau})^T \mathbf{\Sigma}^{-1}_{t+\tau}(\mathbf{y}_{t+\tau} - \boldsymbol{\mu}_{t+\tau})] + ln(2\pi) \qquad (1)$$

where $\mathbf{\Sigma}_t$ is the diagonal covariance matrix and N is the number of samples per batch.

The epistemic uncertainty in the forecast is due to uncertainty on the CNN's weights θ. We recognise that the training dataset (X, Y) is a sample from the true joint distribution of inputs, X, and outputs, Y. We therefore seek the distribution $p(\theta \mid X, Y)$ over θ instead of a single estimate. The Monte-Carlo dropout method approximates $p(\theta \mid X, Y)$ by a parametric distribution $q_\Phi(\theta)$, where Φ is a vector of parameters to tune. Following (Scalia et al., 2019), we model $q_\Phi(\theta)$ as a Bernoulli distribution, β_Φ. In other words, for a given set of input fields, each of the CNN's weights is deactivated with a probability set by the vector Φ, representing the dropout rate of each layer. For the j^{th} parameter this gives $\theta_j \sim \hat{\theta}_j * \beta_{\Phi\, j}$.

Dropout is applied to the network at both training and testing time. During training, dropout prevents overfitting by randomly deactivating some neurons at each epoch. It ensures the predictive capability of the network is distributed across all neurons, instead of converging to a solution in which certain neurons dominate. During testing, we use dropout to produce M Monte-Carlo forecasts, $(\theta^{(i)}, \boldsymbol{\mu}^{(i)}_{t+\tau}, \boldsymbol{\sigma}^{(i)\,2}_{a\;t+\tau})$. We chose $M = 10$ for consistency with the ensemble size of dynamical MJO forecasts, though the computational efficiency of the CNN would enable vastly larger ensemble sizes than this. For the linear layers of the CNN, we apply standard dropout with a dropout rate of 0.3. However, this is not suitable for convolutional layers, because neighbouring points in the feature maps for each layer are often highly correlated (Ghiasi et al., 2018). Instead, we use a DropBlock approach, with a dropout rate of 0.1 for the first convolutional layer and 0.3 for subsequent convolutional layers. In DropBlock, a fraction of points of the maps are randomly set to zero, before all their neighbouring points are also deactivated (Ghiasi

et al., 2018). In this way, DropBlock introduces a correlation between inactive points. However, in contrast to standard dropout, DropBlock disables points on the feature maps and not the weights directly. In Text S6 we demonstrate that deactivating points on the input maps as is carried out in DropBlock is equivalent to the weight deactivation applied in standard dropout, for the case of convolutional layers without bias. This allows us to combine the dropout and DropBlock techniques to represent epistemic uncertainty in the CNN.

Finally, the estimated aleatoric and epistemic uncertainties are combined to give the final predicted mean and total variance:

$$\boldsymbol{\mu}_{t+\tau} = \frac{1}{M}\sum_i \boldsymbol{\mu}^{(i)}_{t+\tau} \tag{2}$$

$$\boldsymbol{\sigma}^2_{tot\ t+\tau} = \frac{1}{M}\sum_i \boldsymbol{\sigma}^{(i)\ 2}_{a\ t+\tau} + \mathrm{Var}(\boldsymbol{\mu}^{(i)}) \tag{3}$$

The network's architecture is shown in supporting figure S2. Our network has three convolutional layers without bias. For all of them, we used Leaky ReLU with $\alpha = 0.003$ as activation function to avoid vanishing gradients. Each of the two first convolutional layers have a (5,5) kernel and are followed by average pooling with a (3,3) kernel size and a (2,1) stride. The third convolutional layer has a (3,3) kernel size. Convolutional layers are followed by two fully-connected layers with 1920 and 200 neurons. The output layer has 4 neurons: the forecast means and aleatoric variances of RMM1 and RMM2. To ensure the output variances are positive, we apply the function $f : x \mapsto \log(1 + \exp(x))$ which we found more stable than ReLU. We train the network with batches of 50 samples up to 35 epochs. Given the large amount of data required to train a deep learning model,

the amount of data needed to make reasonable comparisons with dynamic models and the fact that the Outgoing Longwave Radiation data goes back to 1979 at the most, we made the choice to keep only one train set and one test set and to tune the network parameters (kernel, strides, etc.) on the train set. In order to prevent the overfitting that could result from this choice, we used an L2 regularization in addition to dropout and DropBlock. We observed some sensitivity of the network with respect to the regularization coefficient λ on the train set performance. Hence to avoid overoptimistic results as much as possible, we kept $\lambda = 0.01$, the L2 coefficient that had the best performance on the train set amongst the values of λ high enough to prevent overfitting.

Text S4. *PatternNet*

PatternNet propagates the estimated signal from the output to the input space. Instead of weights, each convolutional or feed-forward layer in PatternNet consists of statistical attribution vectors, which are chosen to maximise certain functions of the covariance between the signal and the output (Kindermans et al., 2017). These vectors are computed layerwise during a training phase, using input fields and corresponding CNN forecasts from the training dataset, and with knowledge of the CNN network weights. Once these vectors are computed, PatternNet is a backpropagation algorithm. The signal s^l at layer l coming from the neuron i is obtained by multiplying the signal s_i^{l+1} of neuron i in the previous layer, $l+1$, with the attribution vector $\mathbf{a}^l$. The signal s_j^l of neuron j in layer l is then the sum of all the signals of its input neurons from layer $l+1$: $s_j^l = \sum_i s_i^{l+1,\,\mathrm{j}}$.

We used the PyTorch implementation of PatternNet by (Translational Neurotechnology Lab, 2019). During backpropagation, when a ReLU layer is encountered, the signal is

backpropagated without modification if the neuron was active during the forward pass and set to zero otherwise. However, the case of Average Pooling and Leaky ReLU layers has not been addressed (Kindermans et al., 2017; Translational Neurotechnology Lab, 2019). For Average Pooling layers the output neuron is an average of input neurons: for such layers we backpropagate the output neuron signal to the inputs without modification. For Leaky ReLU layers, the signal is backpropagated as follows: if the input was positive in the forward pass, the signal is backpropagated without modification, otherwise it is multiplied by the parameter of the Leaky ReLU function (4).

$$s_j^l = \begin{cases} s_j^{l+1} \text{ if positive input in the forward pass} \\ \\ \alpha s_j^{l+1} \text{ otherwise} \end{cases} \tag{4}$$

When using the PatternNet, we use the forecasts of the single CNN member without dropout to simplify the computation. For a given input and corresponding forecast from the CNN, PatternNet provides signals S1 and S2 for each pixel in the input fields, corresponding to RMM1 and RMM2 respectively. The signals S1 and S2 can take any real value. We are interested in signal amplitude and not direction, and so take the absolute value, and then rescale the signals to between 0 and 1. The Signal Mean Maps in Supporting Figures 5–7 are computed with the signals from the test dataset.

Text S5. *Validation Metrics*

CNN and S2S dynamical model forecasts were validated using days with initial observed amplitude above 1.0 (Lim et al., 2018). Three deterministic metrics were considered. The Root Mean Square Error between the forecast and observed RMMs is defined as

$$RMSE(\tau) = \sqrt{\frac{1}{N}\sum_{t=1}^{N}[(f_1(t,\tau) - v_1(t))^2 + (f_2(t,\tau) - v_2(t))^2]} \tag{5}$$

where f_1, f_2 are the forecast mean RMM indices for start date t at lead time τ, v_1, v_2 are the verification RMM indices at that time, and N is the total number of start dates.

The MJO amplitude is defined as

$$A(t,\tau) = \sqrt{(RMM_1(t,\tau)^2 + RMM_2(t,\tau)^2)} \tag{6}$$

The MJO Bivariate Correlation (Supplementary Figure S4) is defined as

$$BV(t,\tau) = \frac{\sum_{t=1}^{N} f_1(t,\tau)v_1(t) + f_2(t,\tau)v_2(t)}{\sqrt{\sum_{t=1}^{N}(f_1(t,\tau)^2 + f_2(t,\tau)^2)} + \sqrt{\sum_{t=1}^{N} v_1(t)^2 + v_2(t)^2}} \tag{7}$$

The amplitude error can then be written

$$ERR_A = \frac{1}{N}\sum_{t=1}^{N}(A_f - A_v) \tag{8}$$

where A_f and A_v are the forecast and verification amplitudes respectively.

Following (Kim et al., 2018), the MJO phase is defined as

$$ERR_P = \frac{1}{N}\sum_{i=1}^{N} \text{atan}(\frac{v_1(t)f_2(t,\tau) - v_2(t)f_1(t,\tau)}{v_1(t)f_1(t,\tau) + v_2(t)f_2(t,\tau)})) \tag{9}$$

Three further scoring rules were used to assess the probabilistic skill of the forecasts. The Continuous Ranked Probability Score (CRPS, (Hersbach, 2000)) is widely used to validate ensemble forecasts.

$$\text{CRPS}(P_f, v) = \int_{-\infty}^{\infty} \left[P_f(x) - \Theta(x - v)\right]^2 \mathrm{d}x, \tag{10}$$

where P_f is the forecast cumulative distribution function, and $\Theta(x - v)$ is the observed cumulative distribution function, which is equal to the Heaviside step function centres on the verification, v. Gaussianity is assumed for forecasts. Following Marshall et al. (2016),

the CRPS of a given day is computed as the sum of the CPRS for RMM1 and RMM2 separately. Then the resulting CRPS are averaged across the whole dataset.

For consistency with the loss function, the log-score, or Ignorance score (Roulston & Smith, 2002), score (equation (1)) was also used to assess forecast skill, where the number of samples N was the number of days in the test data set.

To assess the ability of forecasts to discern predictable from unpredictable days, we compute Error-Spread diagrams following (Leutbecher & Palmer, 2008). For each day we have a data triplet consisting of a predicted mean, variance, and an observed value for RMM1 and RMM2 respectively. We first sort the triplets according to the predicted variance into 5 equally-populated bins. Then for each bin we compute the root mean variance, and the root mean-squared error between the forecast mean and the observation. This process is repeated for RMM1 and RMM2 separately, for each forecast lead time, and for each type of uncertainty (epistemic, aleatoric, and total). For well calibrated forecasts, the average RMSE in each bin should equal the root mean variance.

We also compute the confidence curve $C(\alpha)$ and Error-Drop for each model. For each lead time and each RMM index in turn, we remove the $\alpha\%$ most uncertain cases, and compute the RMSE between the forecast mean and the observed RMM index for the remaining data. We repeat this process setting α to be each of the 20 evenly-spaced quantiles of the RMSE in turn. The Error-Drop is computed from the confidence curve as:

$$\text{Error-Drop} := \frac{C(\alpha_{max})}{C(\alpha_{min})}, \tag{11}$$

where α_{min} and α_{max} correspond to the minimum and maximum fraction of days removed respectively. We set $\alpha_{min} = 0.0$ and $\alpha_{max} = 0.95$.

Text S6. *Monte-Carlo Dropout for DropBlock*

Here, we prove that we can use the Monte-Carlo Dropout method with DropBlock. In our model, the DropBlock is applied after each convolutional layer and these layers do not have bias. Considering a specific convolutional layer layer l, if we denote X the input of this layer, (Gal, 2016) showed that the convolutional operation could be seen as a matrix multiplication $W^T X$ where W is a convolutional weight matrix, rewritten to match the matrix multiplication operation. If we denote m the number of lines of W^T, we can rewrite $W^T X$ with the dot products $W^T X = (W_1^T X, W_2^T X, ..., W_m^T X)^T$.

As we have considered the convolutional operation as a matrix multiplication, $W^T X$ is a column vector of size m. We must rewrite this vector as an output feature map of shape $n \times p$ denoted F, such that $n * p = m$. Each coefficient F_{ij} is equal to a one of the dot products $W_k^T X$. n and p depend on the convolutional parameters (kernel size, stride, dilation).

Then we apply DropBlock. In a first step, each F_{ij} is independently multiplied by a Bernoulli $\beta(p)$. Then in a second time, for each F_{ij}, we consider all its neighbours. We denote d_{ij} the number of neighbours of F_{ij} (in particular, there are less neighbours on the edges than in the center). F_{ij} is disabled if one of its neighbours (or itself) has been disabled during the first step. It is equivalent as considering that F_{ij} has been multiplied by a Bernoulli $\beta(p^{d_{ij}})$. Hence we can write that after the DropBlock,

$$F_{ij}^{*} \;=\; F_{ij}\beta(p^{d_{ij}}) = W_k^T \beta(p^{d_k})X \tag{12}$$

$$W_k^{T\,*} \;=\; W_k^T \beta(p^{d_k}) \tag{13}$$

Thus we conclude that DropBlock applied after a convolutional layer without bias is equivalent to a standard Dropout with a distinct dropout probability for each weight, which can be achieved using the Monte-Carlo Dropout method.

Notes

1. Note that even if the network were supplied with the highest resolution observational data available, these estimates of the observed Earth System would have a finite resolution and would contain errors, thus aleatoric uncertainty remains.

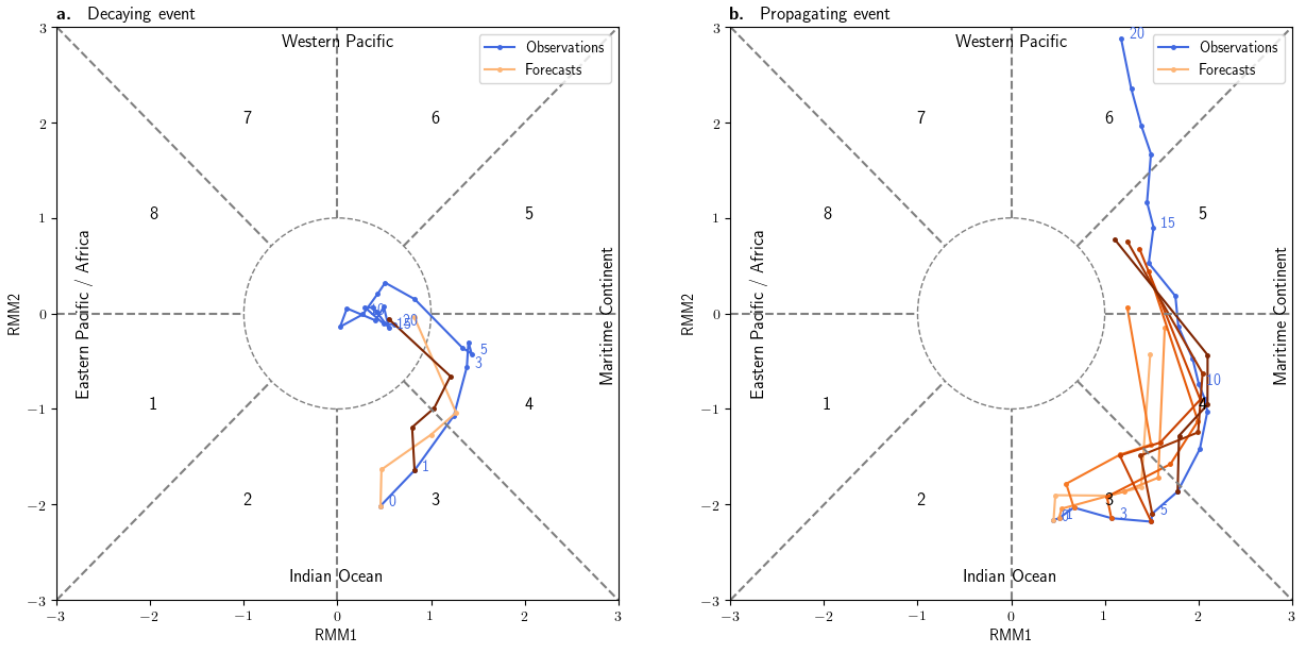

Figure S1. Phase diagrams for a decaying and a propagating event for forecasts initialised in phase 3.

Observations from the first day - 25/02/2006 (**a.**) and 28/02/2012 (**b.**) - are represented up to day-10 in blue. All forecasts (day-1, 3, 5, 10) which began in initial observed phase 3 for each chosen event are represented in shades of orange.

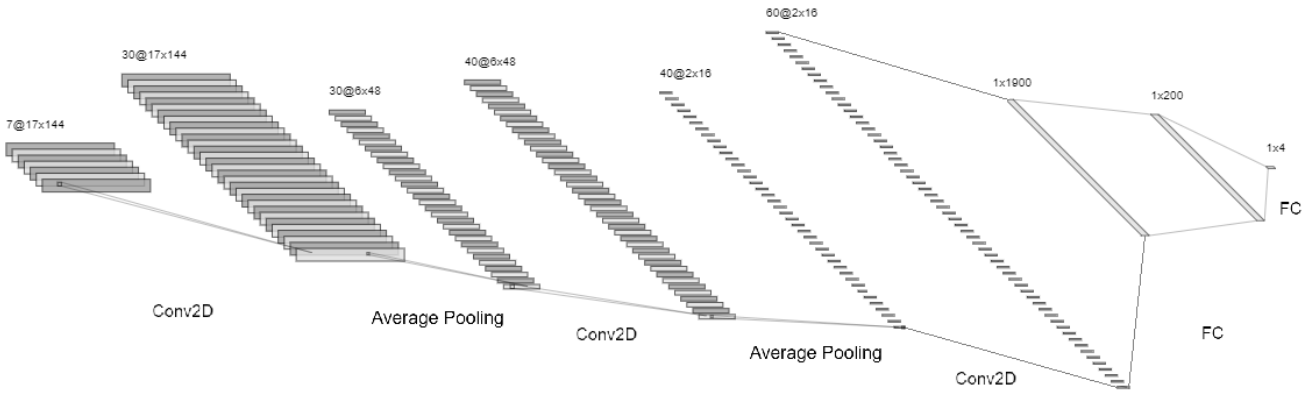

Figure S2. CNN Architecture. Leaky ReLU was used as the activation function of the convolutional layers 1 and 2.

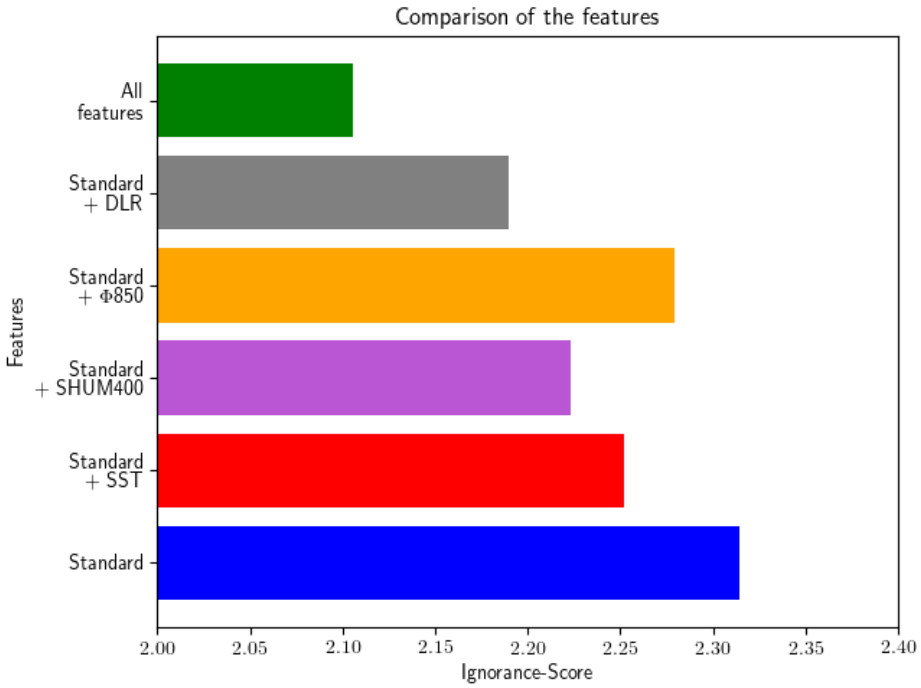

Figure S3. Comparison of the features' performance. Log-score is computed for a CNN trained on different subsets of input features for day-10 forecasts. Days used have initial amplitude above 1.0. Standard stands for "UA200 + UA850 + OLR".

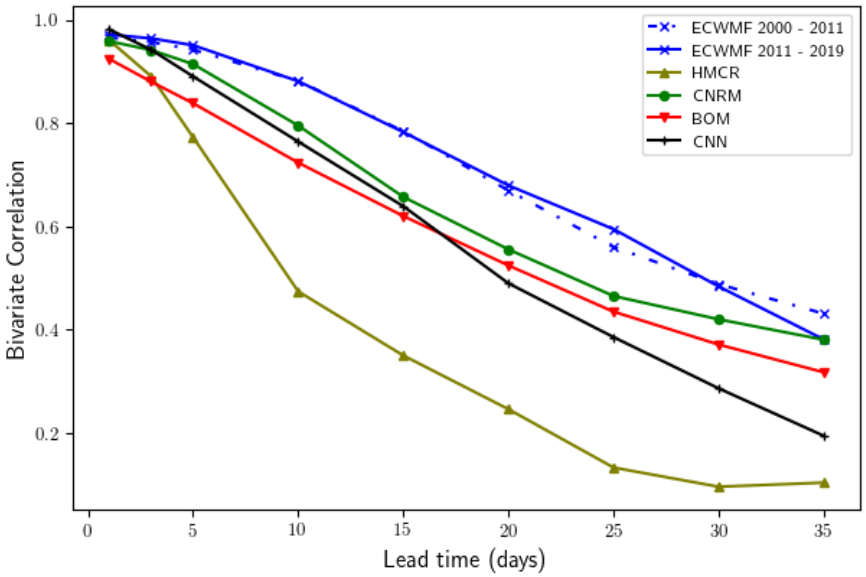

Figure S4. Bivariate correlation computed for RMM1 and RMM2 as a function of lead time. Note that forecasts from different models cover different dates: ECMWF 2000-2019; HMCR 1985-2010; CNRM 1993-2017; BOM 1982-2013; CNN 2011-2019. The ECMWF data was split into two periods to allow direct comparison with the CNN over 2011-2019, and to give an indication of sampling uncertainty.

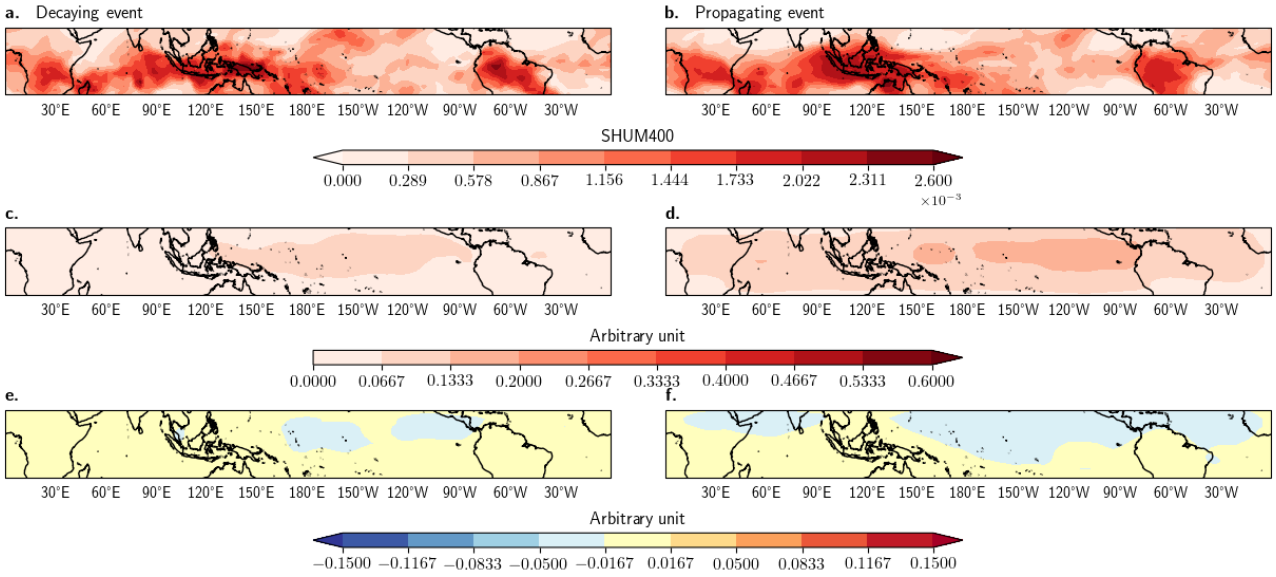

Figure S5. Interpretation of the CNN mean forecasts. (a–b) Composite maps of phase-3 SHUM400 for an MJO event which (a) decays and (b) propagates over the Maritime Continent. (c–d) PatternNet **RMM2** signal mean maps (signal maps averaged over all variables) corresponding to ten-day CNN forecasts for the decaying and propagating event respectively. (e–f) **RMM2** signal anomalies in SHUM400 for the decaying and propagating events respectively. The signal anomalies show a greater focus over the Maritime Continent region for this input variable.

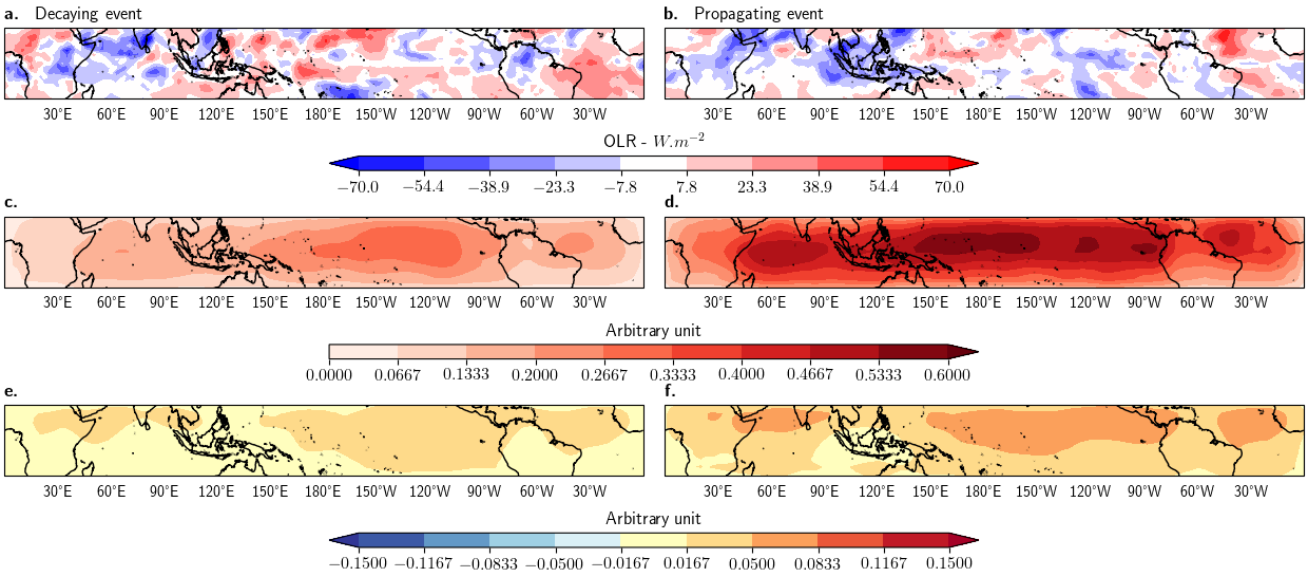

Figure S6. Interpretation of the CNN mean forecasts. (a–b) Composite maps of phase-3 OLR for an MJO event which (a) decays and (b) propagates over the Maritime Continent. (c–d) PatternNet **RMM1** signal mean maps (signal maps averaged over all variables) corresponding to ten-day CNN forecasts for the decaying and propagating event respectively. (e–f) **RMM1** signal anomalies in OLR for the decaying and propagating events respectively. The signal anomalies show a greater focus over the Maritime Continent region for this input variable.

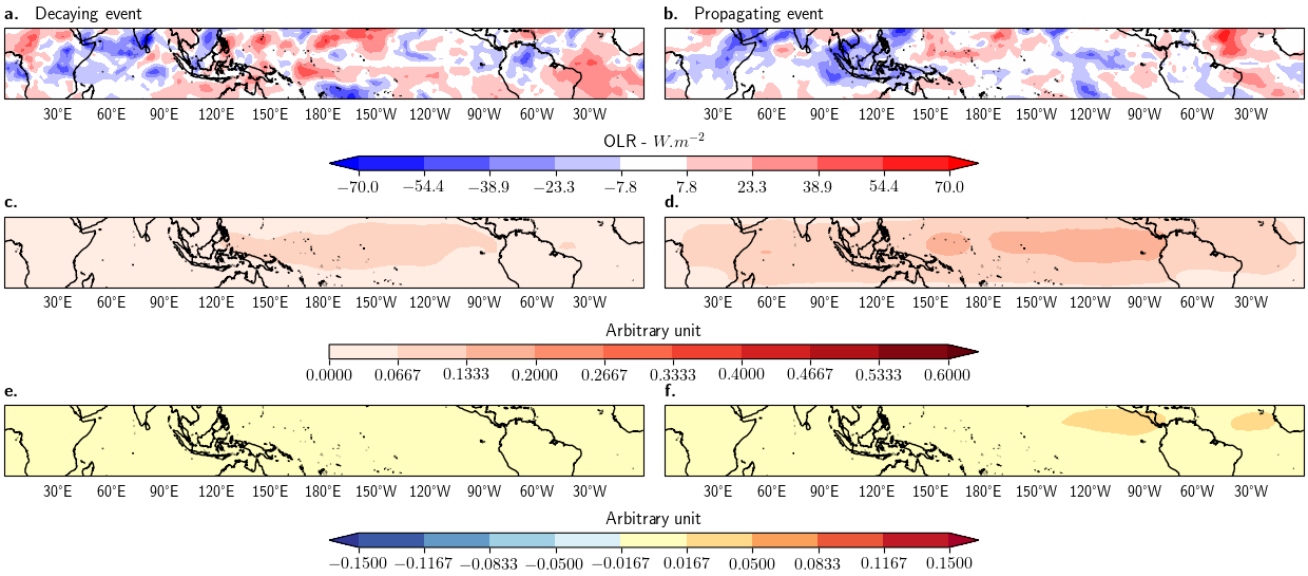

Figure S7. Interpretation of the CNN mean forecasts. (a–b) Composite maps of phase-3 OLR for an MJO event which (a) decays and (b) propagates over the Maritime Continent. (c–d) PatternNet **RMM2** signal mean maps (signal maps averaged over all variables) corresponding to ten-day CNN forecasts for the decaying and propagating event respectively. (e–f) **RMM2** signal anomalies in OLR for the decaying and propagating events respectively. The signal anomalies show a greater focus over the Maritime Continent region for this input variable.

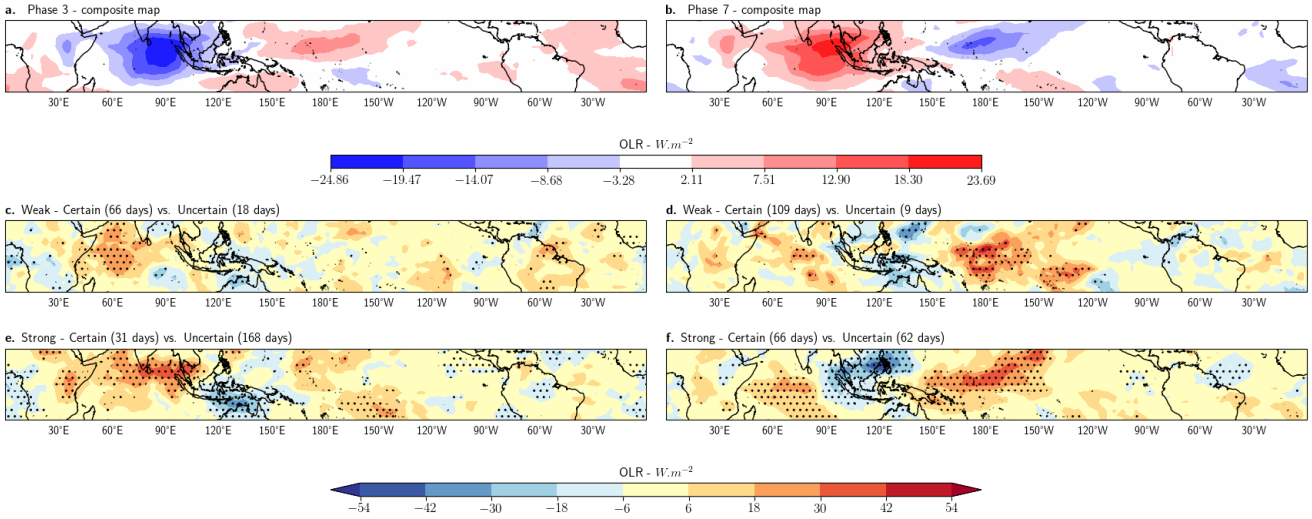

Figure S8. OLR uncertainty interpretation of the CNN MJO forecasts. **a.** and **b.** Composite maps of OLR in initial phases 3 and 7 for day-10 forecasts. Maps have been rescaled using MinMax scaling at each grid point before being fed to the CNN. **c. to f.** Anomalies maps between Weak (Strong) Predictable minus Weak (Strong) Unpredictable events. Weak events have an amplitude below (above) 1.0. Predictable (Unpredictable) events have RMM1 and RMM2 aleatoric uncertainties both inferior (superior) to their 30% (70%) percentiles. Stippling denotes areas where anomalies are significant at the 95% level using the Student's t-test.

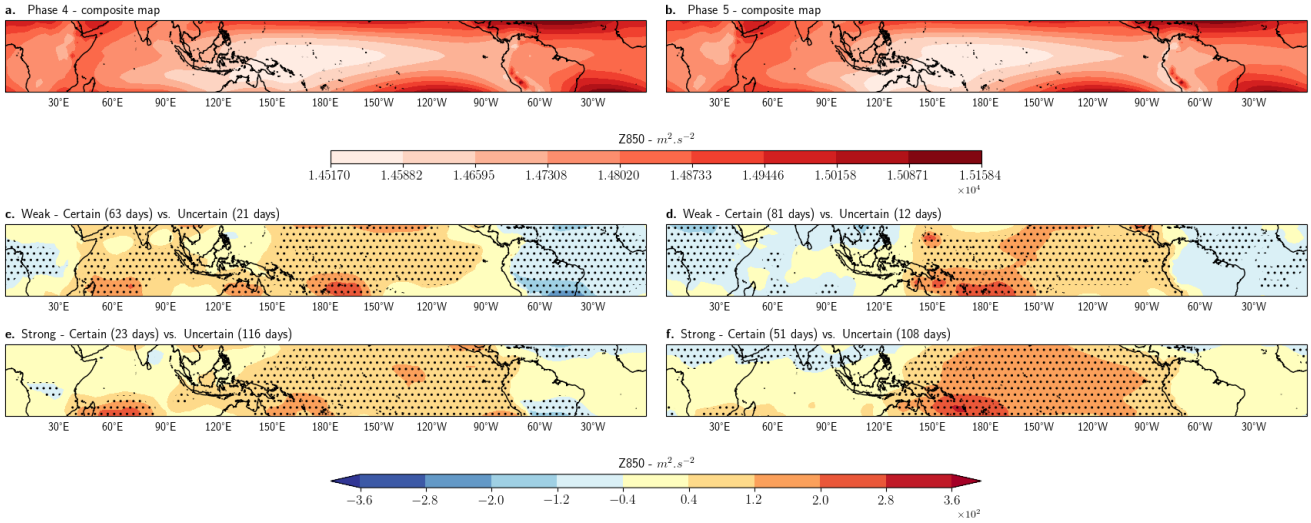

Figure S9. Z850 uncertainty interpretation of the CNN MJO forecasts. **a.** and **b.** Composite maps of geopotential at 850hPa (Z850) in initial phases 4 and 5 for day-10 forecasts. Maps have been rescaled using MinMax scaling at each grid point before being fed to the CNN. **c. to f.** Anomalies maps between Weak (Strong) Predictable minus Weak (Strong) Unpredictable events. Weak events have an amplitude below (above) 1.0. Predictable (Unpredictable) events have RMM1 and RMM2 aleatoric uncertainties both inferior (superior) to their 30% (70%) percentiles. Stippling denotes areas where anomalies are significant at the 95% level using the Student's t-test.

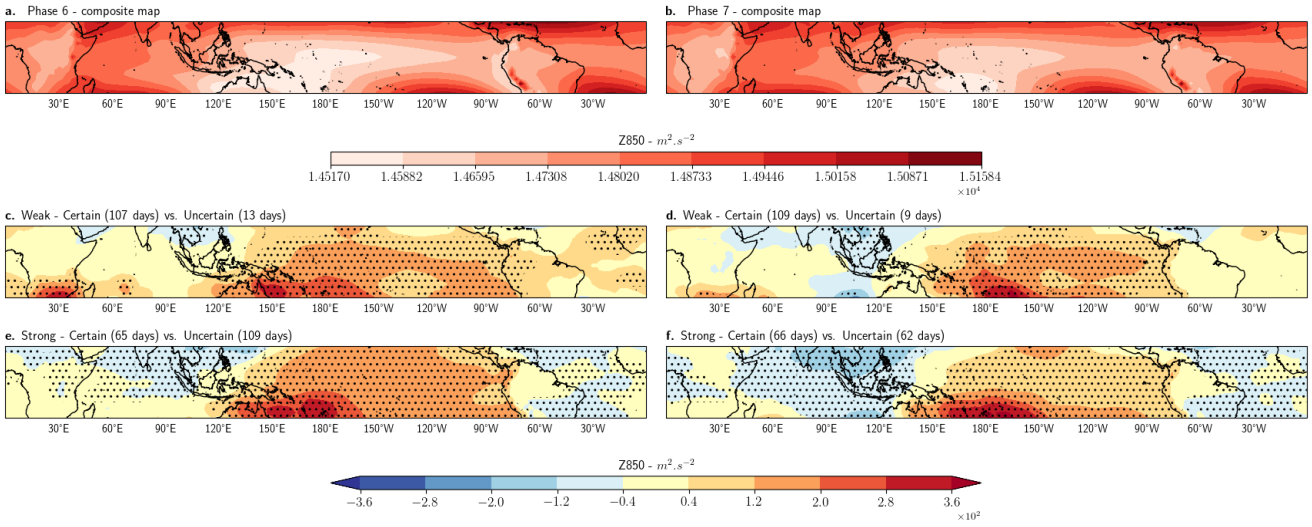

Figure S10. Z850 uncertainty interpretation of the CNN MJO forecasts. **a.** and **b.** Composite maps of geopotential at 850hPa (Z850) in initial phases 6 and 7 for day-10 forecasts. Maps have been rescaled using MinMax scaling at each grid point before being fed to the CNN. **c.** to **f.** Anomalies maps between Weak (Strong) Predictable minus Weak (Strong) Unpredictable events. Weak events have an amplitude below (above) 1.0. Predictable (Unpredictable) events have RMM1 and RMM2 aleatoric uncertainties both inferior (superior) to their 30% (70%) percentiles. Stippling denotes areas where anomalies are significant at the 95% level using the Student's t-test.

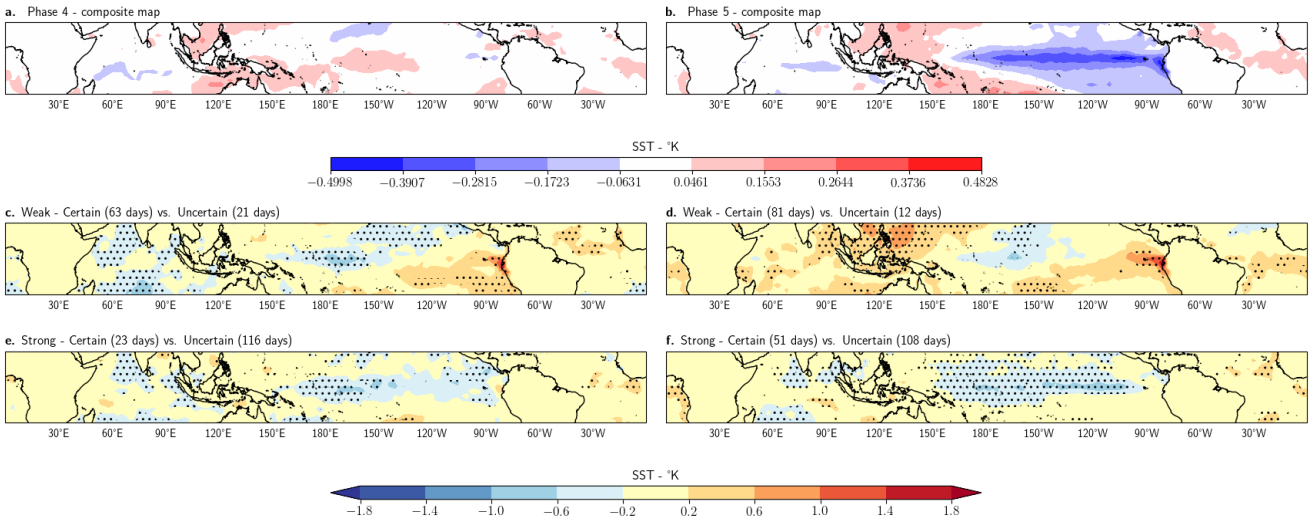

Figure S11. SST uncertainty interpretation of the CNN MJO forecasts. **a.** and **b.** Composite maps of Sea Surface Temperatures (SST) in initial phases 4 and 5 for day-10 forecasts. Maps have been rescaled using MinMax scaling at each grid point before being fed to the CNN. **c. to f.** Anomalies maps between Weak (Strong) Predictable minus Weak (Strong) Unpredictable events. Weak events have an amplitude below (above) 1.0. Predictable (Unpredictable) events have RMM1 and RMM2 aleatoric uncertainties both inferior (superior) to their 30% (70%) percentiles. Stippling denotes areas where anomalies are significant at the 95% level using the Student's t-test.

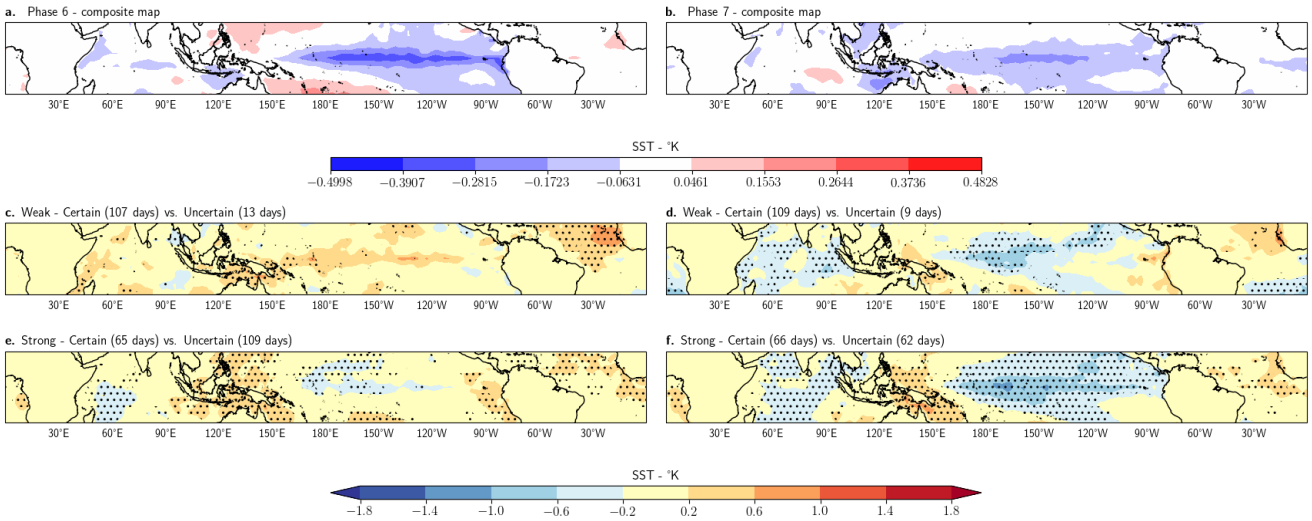

Figure S12. SST uncertainty interpretation of the CNN MJO forecasts. **a.** and **b.** Composite maps of Sea Surface Temperatures anomalies (SST) in initial phases 6 and 7 for day-10 forecasts. Maps have been rescaled using MinMax scaling at each grid point before being fed to the CNN. **c. to f.** Anomalies maps between Weak (Strong) Predictable minus Weak (Strong) Unpredictable events. Weak events have an amplitude below (above) 1.0. Predictable (Unpredictable) events have RMM1 and RMM2 aleatoric uncertainties both inferior (superior) to their 30% (70%) percentiles. Stippling denotes areas where anomalies are significant at the 95% level using the Student's t-test.

Table S1. Description of the dynamical forecast models used for comparison.[a]

Model	Time Range	Reforecast frequency	Ensemble size	Model year
ECMWF	02/01/2000 - 30/11/2019	Twice weekly	11	2020
CNRM	07/01/1993 - 28/12/2017	Weekly	10	2019
BOM	01/01/1982 - 26/12/2013	Twice weekly	33	2014
HMCR	02/01/1985 - 31/12/2010	Weekly	10	2020
CNN (Train)	01/05/1979 - 18/10/2011	Daily	10	2021
CNN (Test)	19/10/2011 - 30/11/2019	Daily	10	2021

[a] The most recent model version were selected according to their availability. The reforecasts are available at `ftp://s2sidx:s2sidx@acquisition.ecmwf.int/RMMS/`

Table S2. All initial phases

	Certain		Uncertain		
	Strong at t	Weak at t	Strong at t	Weak at t	Total
Strong at $t+10$	142	213	707	74	1136
Weak at $t+10$	193	418	117	38	766
Total	335	631	824	112	1902

Table S3. Initial Phase 3

	Certain		Uncertain		
	Strong at t	Weak at t	Strong at t	Weak at t	Total
Strong at $t+10$	16	24	135	13	188
Weak at $t+10$	15	42	33	5	95
Total	31	66	168	18	283

Table S4. Initial Phase 7

	Certain		Uncertain		
	Strong at t	Weak at t	Strong at t	Weak at t	Total
Strong at $t+10$	26	43	56	4	129
Weak at $t+10$	40	66	6	5	117
Total	66	109	62	9	246

References

F. Vitart et al. (2017, January). The Subseasonal to Seasonal (S2S) Prediction Project Database. *Bulletin of the American Meteorological Society*, *98*(1), 163–173. (`ftp://s2sidx:s2sidx@acquisition.ecmwf.int/RMMS`)

Gal, Y. (2016). *Uncertainty in Deep Learning* (Unpublished doctoral dissertation). Cambridge University.

Ghiasi, G., Lin, T.-Y., & Le, Q. V. (2018, October). DropBlock: A regularization method for convolutional networks. *arXiv:1810.12890 [cs]*. (arXiv: 1810.12890)

H. Hersbach et al. (2018a). ERA5 hourly data on pressure levels from 1979 to present. *Copernicus Climate Change Service (C3S) Climate Data Store (CDS)*. (Accessed on 01-03-2021, https://doi.org/10.24381/cds.bd0915c6)

H. Hersbach et al. (2018b). ERA5 hourly data on single levels from 1979 to present. *Copernicus Climate Change Service (C3S) Climate Data Store (CDS)*. (Accessed on 01-03-2021, https://doi.org/10.24381/cds.adbb2d47)

Hersbach, H. (2000). Decomposition of the continuous ranked probability score for ensemble prediction systems. *Weather and Forecasting*, *15*(5), 559–570. doi: 10.1175/1520-0434(2000)015⟨0559:DOTCRP⟩2.0.CO;2

Kim, H., Vitart, F., & Waliser, D. E. (2018, December). Prediction of the Madden–Julian Oscillation: A Review. *Journal of Climate*, *31*(23), 9425–9443. Retrieved from https://journals.ametsoc.org/doi/10.1175/JCLI-D-18-0210.1 doi: 10.1175/JCLI-D-18-0210.1

Kindermans, P.-J., Schütt, K. T., Alber, M., Müller, K.-R., Erhan, D., Kim, B., & Dähne, S. (2017, October). Learning how to explain neural networks: PatternNet and PatternAttribution. *arXiv:1705.05598 [cs, stat]*. Retrieved 2021-05-18, from http://arxiv.org/abs/1705.05598 (arXiv: 1705.05598)

Leutbecher, M., & Palmer, T. (2008, March). Ensemble forecasting. *Journal of Computational Physics*, *227*(7), 3515–3539. Retrieved 2021-05-27, from https://linkinghub.elsevier.com/retrieve/pii/S0021999107000812 doi: 10.1016/j.jcp.2007.02.014

Lim, Y., Son, S.-W., & Kim, D. (2018, May). MJO Prediction Skill of the Subseasonal-to-Seasonal Prediction Models. *Journal of Climate*, *31*(10), 4075–4094. Retrieved 2021-05-05, from http://journals.ametsoc.org/doi/10.1175/JCLI-D-17-0545.1 doi: 10.1175/JCLI-D-17-0545.1

Marshall, A. G., Hendon, H. H., & Hudson, D. (2016). Visualizing and verifying probabilistic forecasts of the Madden-Julian Oscillation. *Geophysical Research Letters*, *43*(23), 12,278–12,286. doi: 10.1002/2016GL071423

Roulston, M. S., & Smith, L. A. (2002). Evaluating probabilistic forecasts using information theory. *Monthly Weather Review*, *130*(6), 1653–1660. doi: 10.1175/1520-0493(2002)130⟨1653:EPFUIT⟩2.0.CO;2

Scalia, G., Grambow, C. A., Pernici, B., Li, Y.-P., & Green, W. H. (2019, October). Evaluating Scalable Uncertainty Estimation Methods for DNN-Based Molecular Property Prediction. *arXiv:1910.03127 [cs, stat]*. (arXiv: 1910.03127)

Translational Neurotechnology Lab. (2019). PatternNet GitHub Repository. (University of Freiburg, https://github.com/TNTLFreiburg/pytorch_patternnet)

Wheeler, M. C., & Hendon, H. H. (2004). An All-Season Real-Time Multivariate MJO Index: Development of an Index for Monitoring and Prediction. *Monthly Weather Review*, *132*, 16.