https://doi.org/10.5194/egusphere-2023-350
Preprint. Discussion started: 17 April 2023

Machine Learning for numerical weather and climate modelling: a review

Catherine O. de Burgh-Day[1] & Tennessee Leeuwenburg[1]

[1]The Bureau of Meteorology, 700 Collins St Docklands, Victoria, Australia

Correspondence to: Catherine O. de Burgh-Day (catherine.deburgh-day@bom.gov.au)

Abstract.

Machine learning (ML) is increasing in popularity in the field of weather and climate modelling. Applications range from improved solvers and preconditioners, to parametrisation scheme emulation and replacement, and recently even to full ML-based weather and climate prediction models. While ML has been used in this space for more than 25 years, it is only in the last 10 or so years that progress has accelerated to the point that ML applications are becoming competitive with numerical knowledge-based alternatives. In this review, we provide a roughly chronological summary of the application of ML to aspects of weather and climate modelling from early publications through to the latest progress at the time of writing. We also provide an overview of key ML concepts and terms. Our aim is to provide a primer for researchers and model developers to rapidly familiarize and update themselves with the world of ML in the context of weather and climate models.

1. Introduction

Current state-of-the-art weather and climate models use numerical methods to solve equations representing the dynamics of the atmosphere and ocean on meshed grids. The grid-scale effects of processes that are too small to be resolved are either represented by parametrization schemes or are prescribed. These numerical weather and climate forecasts are computationally costly and are not amenable to transfer to specialized compute resources such as GPUs. One of the main approaches to improving forecast accuracy is to increase model resolution (reduced timestep between model increments and/or decreased grid spacing), but due to the high computational cost of this approach, improvements in model skill are hampered by the finite supercomputer capacity available. An additional pathway to improve skill is to improve the representation of sub grid-scale processes, however this is again a computationally costly exercise.

Machine learning is an increasingly powerful and popular tool. It has proven to be computationally efficient, as well as being an accurate way to model sub-grid scale processes. The term "Machine learning" (ML) was first coined by Arthur Samuel in 1952 to refer to a "field of study that gives computers the ability to learn without being explicitly programmed"1. Learning by example is the defining characteristic of ML.

[1] http://infolab.stanford.edu/pub/voy/museum/samuel.html, accessed 7th February 2023

https://doi.org/10.5194/egusphere-2023-350
Preprint. Discussion started: 17 April 2023

The growing potential for ML in weather and climate modelling is being increasingly recognized by meteorological agencies and researchers around the world. The former is evidenced by the development of strategies and frameworks to better support the development of ML research, such as the Data Science Framework recently published by the Met Office in the UK[2]. The latter is made clear by the explosion in publications from academia, government agencies and private industry in this space, as demonstrated by the rest of this review.

There are established techniques and aspects of the weather and climate modeling lifecycle that would already be considered ML by many. For example, linear regression†[3], principal component analysis, correlations, and the calculation of teleconnections can all be considered types of ML. Data Assimilation techniques could also be considered a form of ML. There are, however, other classes of ML (e.g. Neural Networks†, Decision Trees†, etc.) which are much less widely used within the weather and climate modelling space and have great potential to be of benefit. There is growing interest in, and increasingly effective application of, these ML techniques to take the place of more traditional approaches to modelling. The potential for ML in weather and climate modelling extends all the way from replacement of individual sub-components of the model (to improve accuracy and reduce computational cost) to full replacement of the entire numerical model.

While ML models are typically computationally costly during training, they can provide very fast predictions at inference† time, especially on GPU hardware. They often also avoid the need to have full understanding of the processes being represented and can learn and infer complex relationships without any need for them to be explicitly encoded. These properties make ML an attractive alternative to traditional parametrization, numerical solver, and modelling methods.

Neural Networks (NNs, explained further in Section 2.1) in particular are an increasingly favored alternative approach for representing sub-grid scale processes or replacing numerical models entirely. They consist of several interconnected layers of nonlinear nodes†, with the number of intermediate layers depending on the complexity of the system being represented. These nodes allow for the encoding of an arbitrary number of interrelationships between arbitrary parameters to represent the system, removing the need to explicitly encode these interrelationships into a parameterization or numerical model.

One challenge that must be overcome before there will be more widespread acceptance of ML as an alternative to traditional modelling methods is that ML is seen as lacking interpretability. Most ML models do not explicitly represent the physical processes they are simulating, although physics constrained ML is a new and growing field which goes some way to addressing this (see Section 6). Furthermore, the techniques available to gain insight into the relative importance and predictive mechanism of each predictor (i.e. the model outputs) are limited. In contrast, traditional models are usually driven by some understanding of the physical mechanisms and processes which are occurring. This makes it possible to more easily gain insight into what physical drivers could explain a given output.

[2] https://www.metoffice.gov.uk/research/foundation/informatics-lab/met-office-data-science-framework, accessed 7 February 2023
[3] Henceforth, the first occurrence of each term described in the glossary is marked with the symbol "†".

https://doi.org/10.5194/egusphere-2023-350
Preprint. Discussion started: 17 April 2023

The "black box" nature of many ML approaches to parametrization makes them an unpopular choice for many researchers, and can be off-putting for decision makers, since, for example, explaining what went wrong in a model after a bad forecast can be more challenging if there are processes in the model which are not, and cannot, be understood through the lens of physics. However, increasing attention is being paid to the interpretability of ML models, and there are existing methods to provide greater insight into the way physical information is propagated through them (e.g., attention maps, which identify the regions in spatial input data that have the greatest impact on the output field, and ablation studies, which involve comparing reduced data sources and/or models to the original models that have full access to available data, to gain insight into the models).

As with their traditional counterparts, ML-based parametrizations and emulators are typically initially developed in single-column models, aquaplanet configurations, or otherwise simplified models. There are many examples of ML-based schemes which have been shown to perform well against benchmark alternatives in this setting, only to fail to do so in a realistic model setting. A common theme is that these ML schemes rapidly excite instabilities in the model because errors in the ML parametrization can push key parameters outside of the domain of the training data as the overall model is integrated forward in time, leading to rapidly escalating errors and to the model 'blowing up'. Similarly, many ML-based full model replacements perform well for short lead times, only to exhibit model drift and a rapid loss of skill for longer lead times due to rapidly growing errors and the model drifting outside its training envelope.

In recent years, however, progress has been made in developing ML parametrizations which are stable within realistic models (i.e. not toy models, aquaplanets etc.), and ML-based full models which can run stably and skillfully to longer lead times. This is usually achieved through training the model on more comprehensive data, employing ML architectures which keep the model outputs within physically real limits, or imposing physical constraints or conservation rules within the ML architecture or training loss functions†.

There are still challenges and possible limitations to an ML approach to weather and climate modelling. In most cases, a robust ML model or parameterization scheme should be able to:

- remain stable in a full (i.e. non-idealized) model run,
- generalize to cases outside its training envelope,
- conserve energy and achieve the required closures.

Additionally, for an ML approach to be worthwhile it must provide one or more of the following benefits:

- For ML parametrization schemes:
    - a speedup of the representation of a sub-grid scale process vs. when run with a traditional parametrization scheme. This can make the difference between the scheme being cost-effective to run or not - when it is not cost-effective the process usually needs to be represented with a static forcing or boundary condition file,

https://doi.org/10.5194/egusphere-2023-350
Preprint. Discussion started: 17 April 2023

- a speedup of the model vs. when run with traditional parametrization schemes,
- improved representation of sub-grid process(es) over traditional parameterization schemes, as measured by metrics appropriate to the situation,
- improved overall accuracy/skill of the model when run with traditional parametrization schemes,
- insight into physical processes not provided by current numerical models or theory.

- For full ML models:
  - a speedup of the model vs. an appropriate numerical model control,
  - improved overall accuracy/skill of the model vs. an appropriate numerical model control,
  - skillful prediction to greater lead times than an appropriate numerical model control,
  - insight into physical processes not provided by current numerical models or theory

Furthermore, in many cases of ML approaches to weather and climate modelling problems (particularly for full model replacement) the work is led by data scientists and ML researchers with limited expertise in weather and climate model evaluation. This can lead to flawed, misleading or incomplete evaluations. Hewamalage et al. (2022) have sought to rectify this problem by providing a guide to forecast evaluation for data scientists.

The scope of this review is deliberately limited to the application of ML within numerical weather and climate models or for their replacement. This is done to keep the length of this review manageable. ML has enormous utility for other aspects of the forecast value chain such as observation quality assurance, data assimilation, model output postprocessing, forecast/product generation, downscaling, impact prediction, decision support tools, etc. A review of the application of, and progress in, ML in these areas would be of great value but is outside the scope of this review and is left for future work. A brief introduction to key ML architectures and concepts, including suggested foundational reading, is also provided to aid readers who are unfamiliar with the subject.

The remainder of this review is structured as follows: In Section 2 a quick introduction to ML is provided. In Section 3 ML use in sub-grid parametrization and emulation, along with tools and challenges specific to this domain, are covered. In Section 4 the application of ML for the partial differential equations governing fluid flow is reviewed. In Section 5 the use of ML for full model replacement or emulation is reviewed. In Section 6 the growing field of physics constrained ML models is introduced, and in Section 7 a number of topics tangential to the main focus of this review are briefly mentioned. In Section 8 a review of the history of, and progress in, ML outside of the fields of weather and climate science is presented. In Section 9 some practical considerations for the integration of ML innovations into operational and climate models are discussed, and Section 10 provides a summary. A Glossary of Terms is provided after the final Section to aid the reader in their understanding of key concepts and words.

https://doi.org/10.5194/egusphere-2023-350
Preprint. Discussion started: 17 April 2023

2. A Quick Introduction to Machine Learning

While the scope of this paper is a review of ML work directly applicable to weather and climate modelling, an abridged introduction to some key fundamental ML concepts is provided here to aid the reader. Suggested starting points for interested readers include Chase et al (2022a, 2022b), Russell & Norvig (2021), Goodfellow et al. (2016), and Hastie et al. (2009).

This introductory section is a brief exposition of the concepts most central to this review. Definitions for this section can be found in the glossary.

The majority of ML methods which have found traction in weather and climate modelling were first developed in fields such as computer vision, natural language processing and statistical modelling. Few, if any, of the methods mentioned in this paper could be considered unique to weather and climate modelling, however, they have in many cases been modified to a greater or lesser extent to suit the characteristics of the problem. Furthermore, there is a trend towards increasingly customized architectures in this field as it matures.

In this review, the term algorithm refers to the mathematical underpinnings of a machine learning approach. By this definition, decision trees (DTs), NNs, linear regression and Fourier transforms are examples of algorithms. The two most relevant algorithms for this review are DTs and NNs.

The term architecture in machine learning refers to a specific way of utilizing an algorithm to achieve a modelling objective reliably. For example, the U-Net† architecture is a specific way of laying out a NN which has proven effective in many applications. The extreme gradient boosting decision tree† architecture is a specific way of utilizing DTs which has proven reliable and effective for an extraordinary number of problems and situations and is an excellent choice as a first tool to experiment with machine learning.

A major current focus of ML research is new architectures based on NNs. Research also continues into the algorithms themselves. Many other algorithms have been and continue to be employed in machine learning but are not central to this review.

A key point for ML researchers to be aware of is the critical importance of approaching model training carefully. There are many pitfalls which can result in underperformance, unexpected bias or misclassification. For instance, adversarial examples† can occur ‘naturally’, and systems which process data can be subject to adversarial attack† through the intentional supply of data designed to fool a trained network.

2.1. Introduction to Neural Networks

NNs can be regarded as universal function approximators (Hornik et al., 1989; see also Lu et al., 2019). Further, NN architectures can theoretically be themselves modelled as a very wide feed-forward† NN with a single hidden layer. A Fourier transform is another example of a function approximator, although it is not universal since not all functions are periodic. NNs can therefore be candidates for accurate modelling of physical processes.

https://doi.org/10.5194/egusphere-2023-350
Preprint. Discussion started: 17 April 2023

ML models are typically introduced in the literature as being either classification† or regression† models, and either supervised† or unsupervised†.

The mathematical underpinning of a NN can be considered distinctly in terms of its evaluation† (i.e., output, or prediction) step and its training update step. The prediction step can be considered as the evaluation of a many-dimensional arbitrarily complex function.

The simplest NN is a single-input, single node network with a simple activation† function. A commonly used activation function is the sigmoid function, which helpfully compresses the range between 0 and 1. A classification model will employ a threshold to map the output into the target categories. A regression model seeks to optimize the output result against some target value for the function.

Complex NNs are built up from many individual nodes, which may have heterogenous activation functions and a complex connectome†. The forward pass†, by which inputs are fed into the network and evaluated against activation functions to produce the final prediction, uses computationally efficient processes to quickly produce the result.

The training step for a NN is far more complex. The earliest NNs were designed by hand rather than through automation. The training step applies a back-propagation† algorithm to apply adjustment factors to the weights† and biases† of each node based on the accuracy of the overall prediction from the network.

Training very large networks was initially impractical. Both hardware and architecture advances have changed this, resulting in the significant increase in application of NNs to practical problems. Most NN research explores how to utilize different architectures to train more effective networks. There is little research going into improving the prediction step as the effectiveness of a network is limited by its ability to learn rather than its ability to predict. Some research into computational efficiency is relevant to the predictive step. NNs can still be technically challenging to work with, and a lot of skill and knowledge are needed to approach new applications.

The major classes of NN architectures most likely to be encountered are:

- Token-sequence architectures, first applied to natural language processing, generation and translation; applicable to any time-series problem. Now also applied to image and video applications, and mixed-mode applications such as text-to-image or text-to-video
- Convolutional† architectures, first applied to image content recognition, which match the connectome of the network to the fine structure of images in hierarchical fashion to learn to recognize high-level objects in images
- Transformer architectures†, based on the attention mechanism† to provide a non-recurrent architecture which can be trained using parallelized training strategies. This allows larger models to be trained. Originally developed for sequence prediction and extended to image processed through vision transformer architectures.

https://doi.org/10.5194/egusphere-2023-350
Preprint. Discussion started: 17 April 2023

2.2. Introduction to Decision Trees

DTs are a series of decision points, typically represented in binary fashion based on a simple threshold. A particular DT of a particular size maps the input conditions into a final 'leaf' node which represents the outcome of the decisions up to that point.

A random forest[†] (RF) is the composition of a large number of DTs assembled according to a prescribed generation scheme, which are used as an ensemble. A gradient boosted decision tree (GBDT) is built up sequentially, where each subsequent decision tree attempts to model the errors of the stack of trees built up thus far. This approach outperforms RFs in most cases.

The DT family of ML architectures are very easy to train and are very efficient. They are well documented in the public domain and in published literature. DTs are statistical in nature and are not capable of effectively generalizing to situations which are not similar to those seen during training. This can be an advantage when unbounded outputs would be problematic, however can lead to problems where an ability to produce out-of-training solutions is necessary. Additionally, current DT implementations require all nodes (of all trees in the case of RFs and GBDTs) to be held in memory at inference time, making them potentially memory heavy.

3. Sub-grid parametrization and emulation

Sub-grid scale processes in numerical weather and climate models are typically represented via a statistical parameterization of what the macroscopic impacts of the process would be on resolved processes and parameters. These are commonly referred to as parameterization schemes, and can be very complex and relatively computationally costly. For example, in the European Centre for Medium-Range Weather Forecast's (ECMWF) Integrated Forecasting System (IFS) model they account for about a third of the total computational cost of running the model (Chantry et al. 2021b). They also require some understanding of the underlying unresolved physical processes. Examples of sub-grid scale processes which are typically currently parameterized in operational systems include gravity wave drag, convection, radiation, sub-grid scale turbulence, and cloud microphysics. As additional complexity (for example representation of aerosols, atmospheric chemistry, land surface processes, etc.) is added to numerical models, the computational cost will only increase.

ML presents an alternative approach to representing sub-grid scale processes, either by emulating the behavior of an existing parametrization scheme, emulating the behavior of sub-components of the scheme, by replacing the current scheme or sub-component entirely with an ML-based scheme, or by replacing the aggregate effects of multiple parametrization schemes with a single ML model.

ML emulation of existing schemes or sub-components has the advantage of maintaining the status quo within the model; no or minimal re-tuning of the model should be required since the ML emulation is trained to replicate the results of an already-tuned-for scheme. Because of this, the main benefit of this approach is that it reduces the computational cost of running the parametrization scheme. On the other hand, full replacement of an existing

https://doi.org/10.5194/egusphere-2023-350
Preprint. Discussion started: 17 April 2023

parameterization scheme or sub-component with an ML alternative has the potential to be both computationally cheaper and also an improvement over the preceding scheme.

In the following subsections, a review of the literature on aspects of ML for the parametrization and emulation of sub grid-scale processes is presented.

3.1. Early work on ML parametrization and ML emulations

A popular target for applying ML in climate models is radiative transfer, since it is one of the more computationally costly components of the model. As such, many early examples of the use of ML in sub-grid parametrization schemes focus on aspects of this physical process. Chevallier et al. (1998) trained NNs to represent the radiative transfer budget from the top of the atmosphere to the land surface, with a focus on application in climate studies. They incorporated the information from both line-by-line and band models in their training to achieve competitive results against both benchmarks. Their NNs achieved accuracies comparable to or better than benchmark radiative transfer models of the time, while also being much faster computationally.

In contrast to the ML based scheme developed by Chevallier et al. (1998), which could be considered an entirely new parametrization scheme, Krasnopolsky et al. (2005) used NNs to develop an ML based emulation of the existing atmospheric longwave radiation parametrization scheme in the NCAR Community Atmospheric Model (CAM). The authors demonstrated speedups with the NN emulation of 50-80 times the original parameterization scheme.

Emulation of existing schemes has since then become a popular method for achieving significant model speedups. For example, Gettelman et al. (2021) investigated the differences between a GCM with the warm rain formation process replaced with a bin microphysical model (resulting in a 400% slowdown) and one with the standard bulk microphysics parameterization in place. They then replaced the bin microphysical model with a set of NNs designed to emulate the differences observed, and showed that this configuration was able to closely reproduce the effects of including the bin microphysical model, without any of the corresponding slowdown in the GCM.

3.2. ML for coarse graining

Coarse graining involves using higher resolution model or analysis data to map the relationship between smaller-scale processes and a coarser grid resolution. It can be used to develop parameterization schemes without explicitly representing the physics of smaller scale processes.

This has proven to be a popular method for developing ML-based parametrization schemes. Brenowitz & Bretherton (2018) used a near-global aqua planet simulation run at 4 km grid length to train a NN to represent the apparent sources of heat and moisture averaged onto 160 km^2 grid boxes. They then tested this scheme in a prognostic single column model and showed that it performed better than a traditional model in matching the behavior of the aqua planet simulation it was trained on. Brenowitz & Bretherton (2019) built on this work by training their NN on the same global aqua-planet 4 km simulation, but then embedded this scheme within a coarser resolution (160 km^2) global aqua planet General Circulation Model (GCM). Embedding NNs within GCMs is challenging because feedbacks between NN and

https://doi.org/10.5194/egusphere-2023-350
Preprint. Discussion started: 17 April 2023

GCM components can cause spatially extended simulations to become dynamically unstable within a few model days. This is due to the inherently chaotic nature of the atmosphere in the GCM responding to inputs from the NN which cause rapidly escalating dynamical instabilities and/or violate physical conservation laws. The authors overcame this by identifying and removing inputs into the NN which were contributing to feedbacks between the NN and GCM (Brenowitz et al. 2020), and by including multiple time steps in the NN training cost function. This resulted in stable simulations which predicted the future state more accurately than the course resolution GCM without any parametrization of subgrid-scale variability, however the authors do observe that the mean state of their NN-coupled GCM would drift, making it unsuitable for prognostic climate simulations.

Rasp et al. (2018) trained a Deep Neural Network† (DNN) to represent all atmospheric subgrid processes in an aquaplanet climate model by learning from a multiscale model in which convection was treated explicitly. They then replaced all sub-grid parameterizations in an aquaplanet GCM with the DNN, and allowed it to freely interact with the resolved dynamics and the surface-flux scheme. They showed that the resulting system was stable and able to closely reproduce not only the mean climate of the cloud-resolving simulation but also key aspects of variability in prognostic multiyear simulations. The authors noted that their decision to use DNNs was a deliberate one, because they proved more stable in their prognostic simulations than NNs, and they also observed that larger networks achieved lower training losses. However, while Rasp et al. (2018) were able to engineer a stable model that produced results close to the reference GCM, small changes in the training dataset or input and output vectors quickly led to the NN producing increasingly unrealistic outputs and causing model blow-ups (Rasp 2020). Consistent with this, Brenowitz & Bretherton (2019) report that they were unable to achieve the same improvements in stability with increasing network layers found by Rasp et al. (2018).

3.3. Overcoming instability in ML emulations and parametrizations

O'Gorman & Dwyer (2018) tackled the instabilities observed in NN- and DNN-based approaches to subgrid-scale parameterization by employing an alternative ML method; Random Forests (RFs; Breiman 2001; Tibshirani & Friedman 2001). The authors trained a RF to emulate the outputs of a conventional moist convection parametrization scheme. They then replaced the conventional parameterization scheme with this emulation within a global climate model, and showed that it ran stably and was able to accurately produce climate statistics such as precipitation extremes without needing to be specially trained on extreme scenarios. RFs consist of an ensemble of DTs, with the predictions of the RF being the average of the predictions of the DTs which in turn exist within the domain of the training data. RFs thus have the property that their predictions cannot go outside of the domain for their training data, which in the case of O'Gorman & Dwyer (2018) ensured conservation of energy and nonnegativity of surface precipitation (both critically important features of the moist convection parametrization scheme) were automatically achieved. A disadvantage of this method however is that it requires considerable memory when the climate model is being run to store the tree structures and predicted values which make up the RF.

Yuval & O'Gorman (2020) extended on the ideas in O'Gorman & Dwyer (2018), switching from emulation of a single parametrization scheme to emulation of all atmospheric sub grid processes. They trained an RF on a high-resolution

https://doi.org/10.5194/egusphere-2023-350
Preprint. Discussion started: 17 April 2023

three-dimensional model of a quasi-global atmosphere to produce outputs for a course-grained version of the model, and showed that at course resolution the RF can be used to reproduce the climate of the high-resolution simulation, running stably for 1000 days.

There are some drawbacks to a RF approach compared to a NN approach however; namely that NNs provide the possibility for greater accuracy than RFs, and also require substantially less memory when implemented. Given that GCMs are already memory intensive this can be a limiting factor in the practical application of ML parametrization schemes. Furthermore, there is the potential to implement reduced precision NNs on Graphics Processing Units (GPUs) and Central Processing Units (CPUs) which still achieve sufficient accuracy, leading to substantial gains in computational efficiency. Motivated by these considerations, Yuval et al. (2021) trained a NN in a similar manner to how the RF in Yuval & O'Gorman (2020) was trained, using a high resolution aqua-planet model and aiming to coarse grain the model parameters. They overcame the model instabilities observed to occur in previous attempts to use NNs for this process by wherever possible training to predict fluxes and sources and sinks (as opposed to the net tendencies predicted by the RF in Yuval & O'Gorman (2020)), thus incorporating physical constraints into the NN parametrization. The authors also investigated the impact of reduced precision in the NN, and found that it had little impact on the simulated climate.

3.4. From aquaplanets to realistic land-ocean simulations

All of the studies discussed in this section so far which were tested in a full GCM have used aqua planet simulations. Han et al. (2020) broke away from this trend by developing a Residual Neural Network (ResNet) based parametrization scheme which emulated the moist physics processes in a realistic land-ocean simulation. Their emulation reproduced the characteristics of the land-ocean simulation well, and was also stable when embedded in single column models.

Mooers et al. (2021) represents a subsequent example of an ML emulation of atmospheric fields with realistic geographical boundary conditions, where the authors developed feed-forward DNNs to super-parametrize sub grid-scale atmospheric parameters and forced a realistic land surface model with them. Super-parametrization is distinct from traditional parameterization in that it relies on solving (usually simplified) governing equations for sub grid-scale processes rather than heuristic approximations of these processes. They employed automated hyperparameter optimization[†] to investigate a range of neural network architectures across ~250 trials, and investigated the statistical characteristics of their emulations. While the authors found that their DNNs had a less good fit in the tropical marine boundary layer, attributable to the DNN struggling to emulate fast stochastic signals in convection, they also reported good skill for signals on diurnal to synoptic timescales.

Brenowitz et al. (2022) sought to address the challenge of emulating fast processes. They used FV3GFS (Zhou et al., 2019; Harris et al., 2021; a compressible atmospheric model used for operational weather forecasts by the US National Weather Service) with a simple cloud microphysics scheme included to generate training data and used this to train a selection of ML models to emulate cloud microphysics processes, including fast phase changes. They emulated

https://doi.org/10.5194/egusphere-2023-350
Preprint. Discussion started: 17 April 2023

different aspects of the microphysics with separate ML models chosen to be suitable to each task. For example, simple parameters were trained with single-layer NNs, while parameters which are more complex spatially were trained with RNNs (e.g., rain falls downwards and not upwards, so it is sequential in timesteps through the atmosphere – a feature which can be represented by an RNN). They then embedded their ML emulation in FV3GFS. They found that their combined ML simulation performed skillfully according to their chosen metrics, but had excessive cloud over the Antarctic Plateau.

All of these studies, however, did not test their parameterizations in prognostic long-term simulations.

3.5. Testing with prognostic long-term simulations

A barrier to achieving stable runs with minimal model drift with ML components is the fact that generic ML models are not designed to conserve quantities which are required to be conserved by the physics of the atmosphere and ocean. Beucler et al. (2019) proposed and tested two methods for imposing such constraints in a NN model; (1) constraining the loss function or (2) constraining the architecture of the network itself. They found that their control NN with no physical constraints imposed performed well, but did so by breaking conservation laws, bringing into question the trustworthiness of such a model in a prognostic setting. Their constrained networks did however generalize better to unforeseen conditions, implying they might perform better under a changing climate than unconstrained models.

Chantry et al. (2021b) trained a NN to emulate the non-orographic gravity wave drag parameterization in the ECMWF IFS model (specifically cycle 45R1, ECMWF, 2018) and were able to run stable, accurate simulations out to 1 year with this emulation coupled to the IFS. While the authors note that RFs have been shown to be more stable (e.g., O'Gorman & Dwyer (2018) and Yuval & O'Gorman (2020), as described above, and Brenowitz et al. (2020)), they chose to focus on NNs since they have lower memory requirements and therefore promise better theoretical performance. The authors assessed the performance of their emulation in a realistic GCM by coupling the NN with the IFS, replacing the existing non-orographic gravity wave drag scheme, and performed 120 hour, 10 day, and 1 year forecasts at ~25 km resolution in a variety of model configurations. The authors showed that their emulation was able to run stably when coupled to the IFS for seasonal timescales, including being able to reproduce the descent of the Quasi-biennial Oscillation (QBO). Interestingly, while the authors initially aimed to ensure momentum conservation in a manner similar to Beucler at al. (2021), they found that this constraint led to model instabilities and that a better result was achieved without it. One possible explanation for this is that Beucler at al. (2021) assessed their NNs in an aquaplanet setting. Nonetheless, Chantry et al. (2021b) noted that since their method was not identical to Beucler et al. (2021), improved stability could potentially be achieved by following their method more precisely. The computational cost of the NN emulation developed by Chantry et al. (2021b) was found to be similar that of the existing parametrization scheme when run on CPUs, but was faster by a factor of 10 when run on GPUs due to the reduction in data transmission bottlenecks.

The first study to successfully run stable long-term climate simulations with ML parametrizations was Wang et al. (2022a), who extended on the work of Han et al. (2020) by constructing a Residual DNN† (ResDNN) to emulate moist

https://doi.org/10.5194/egusphere-2023-350
Preprint. Discussion started: 17 April 2023

physics processes. They used the residual connections from Han et al. (2020) to construct DNNs with good nonlinear fitting ability, and filtered out unstable NN parametrizations using a trial-and-error analysis, resulting in the best ResDNN set in terms of accuracy and long-term stability. They implemented this scheme in a GCM with realistic geographical boundary conditions and were able to maintain stable simulations for over 10 years in an Atmospheric Model Intercomparison Project (AMIP)-style configuration. This was more akin to a hybrid ML-physics based model than a traditional GCM with ML-based parametrization, because rather than embedding the ResDNN in the model code, the authors used a NN-GCM coupling platform through which the NNs and GCMs could interact through data transmission. This is in contrast to the approach employed in the Physical-model Integration with Machine Learning[4] (PIML) project and Infero[5], which are both described in Section 3.11. One advantage to this approach noted by the authors is that it allows for a high degree of flexibility in the application of the ML component, however is likely to be less efficient than a fully-embedded ML model, due to the potential for data transmission bottlenecks.

3.6. Training with observational data

An alternative to using more complex and/or higher resolution models for training data is to train using direct observational data. For example, Ukkonen & Mäkelä (2019) used reanalysis data from ERA5 and lightning observation data to train a variety of different types of ML models to predict thunderstorm occurrence; this was then used as a proxy to trigger deep convection. ML models assessed were logistic regression, RFs, GBDTs, and DNNs, with the final two showing a significant increase in skill over convective available potential energy (CAPE; a standard measure of potential convective instability). One of the challenges of accurately reproducing the large-scale effects of convection is correctly identifying when deep convection should occur within a grid cell. The authors proposed that an ML model such as those they assessed could be used as the “trigger function” which activates the deep convection scheme within a GCM.

3.7. ML for super parameterization

Revisiting the topic of super parametrized sub grid scale processes introduced above, the use of ML for this approach was investigated in depth by Chattopadhyay et al. (2020). The authors introduced a framework for NN-based super parametrization, and compared the performance of this method against NN-based traditional parametrization (i.e., based on heuristic approximations of sub grid-scale processes) and direct super parameterization (i.e., explicitly solving for the sub grid-scale processes) in a chaotic Lorenz '96 (Lorenz 1996) system that had three sets of variables, each of a different scale. They found that their NN-based super parameterization outperformed direct super parameterization in terms of computational cost, and was more accurate than NN-based traditional parametrization. The NN-based super parameterization showed comparable accuracy to direct super parameterization in reproducing long-term climate statistics, but was not always comparable for short-term forecasting.

[4] https://turbo-adventure-f9826cb3.pages.github.io/, accessed 7th February 2023
[5] https://infero.readthedocs.io/en/latest/, accessed 7th February 2023

https://doi.org/10.5194/egusphere-2023-350
Preprint. Discussion started: 17 April 2023

3.8. Stochastic parametrization schemes

A more recent approach to the representation of sub grid-scale processes is via stochastic parameterization schemes, which can represent uncertainty within the scheme. There has been less focus on replacing these schemes with ML alternatives than non-stochastic schemes, however some progress has been made. Krasnopolsky et al. (2013) used an ensemble of NNs to learn a stochastic convection parametrization from data from a high-resolution cloud resolving model. In this case, the stochastic nature of the parametrization was captured by the ensemble of NNs. Gagne et al (2020) took a different approach, investigating the utility of generative adversarial networks (GANs) for stochastic parametrization schemes in Lorenz '96 (Lorenz 1996) models. In this case, the GAN learned to emulate the noise of the scheme directly, rather than implicitly representing it with an ensemble. They described the effects of different methods to characterize input noise for the GAN, and the performance of the model at both weather and climate timescales. The authors found that the properties of the noise influenced the efficacy of training. Too much noise resulted impaired model convergence and too little noise resulted in instabilities within the trained networks.

3.9. ML parametrization and emulation for ocean models

ML approaches to parameterization of subgrid-scale processes is not limited to the atmosphere. Krasnopolsky et al. (2002) presented an early application of NN for the approximation of seawater density, the inversion of the seawater equation of state, and a NN approximation of the nonlinear wave-wave interaction. More recently, Bolton & Zanna (2019) investigated the utility of Convolutional Neural Networks (CNNs) for parametrizing unresolved turbulent ocean processes and subsurface flow fields. Zanna & Bolton (2020) then investigated both Relevance Vector Machines† (RVMs) and CNNs for parameterizing mesoscale ocean eddies. They demonstrate that because RVMs are interpretable, they can be used to reveal closed-form equations for eddy parameterizations with embedded conservation laws. The authors tested the RVM and CNN parameterizations in an idealized ocean model and found that both improved the statistics of the coarse resolution simulation. While the CNN was found to be more stable than the RVM, the advantage of the RVM is the greater interpretability of its outputs.

3.10. ML for representing or correcting a sub-component of a parametrization scheme

An alternative method to replacing or emulating an entire parametrization scheme or schemes with ML is to target the most costly or troublesome sub-components of the scheme, and either replace those or make corrections to them.

Ukkonen et al. (2020) trained NNs to replace gas optics computations in the RTE-RRTMGP (Radiative Transfer for Energetics and Rapid and accurate Radiative Transfer Model for General circulation models applications-Parallel; Pincus et al., 2019) scheme. The NNs were faster by a factor of 1-6, depending on the software and hardware platforms used. The accuracy of the scheme remained similar to that of the original scheme.

Meyer et al. (2022) trained a NN to account for the differences between 1D cloud effects in the European Centre for Medium Range Weather Forecasting (ECMWF) 1D radiation scheme ecRad and 3D cloud effects in the ECMWF SPARTACUS (SPeedy Algorithm for Radiative TrAnsfer through CloUd Sides) solver. The 1D cloud effects solver

https://doi.org/10.5194/egusphere-2023-350
Preprint. Discussion started: 17 April 2023

within ecRad, Tripleclouds, is favored over the 3D SPARTACUS solver because it is five times less computationally expensive. The authors show that their NN can account for differences between the two schemes with typical errors between 20% and 30% of the 3D signal, resulting in an improvement in Tripleclouds' accuracy with an increase in runtime of approximately 1%. By accounting for the differences between SPARTACUS and Tripleclouds rather than emulating all of SPARTACUS, the authors were able to keep Tripleclouds unchanged within ecRad for cloud-free areas of the atmosphere, and utilize the NN 3D correction elsewhere.

3.11. Bridging the gap between popular languages for ML and large numerical models

A common toolset for researchers to develop and experiment with different ML approaches to problems is Python libraries such as pytorch, scikit-learn, tensorflow, keras, etc., or other dynamically-typed, non-precompiled languages. In contrast, numerical weather models are almost universally written in statically-typed compiled languages, predominantly Fortran. To make use of ML emulations or parameterizations in the models thus requires that they be:

(1) treated as a separate model periodically coupled to the main model (as is done between atmosphere and ocean models for example), or
(2) be manually re-implemented in Fortran, or
(3) that the pre-existing libraries used are somehow be made accessible within the model code.

Wang et al. (2022a; mentioned already above) opted for method 1, developing what could be considered a hybrid ML-physics based model rather than a traditional GCM with ML-based parametrization. In their study, the authors used a NN-GCM coupling platform through which the NNs and GCMs could interact through data transmission. One advantage to this approach noted by the authors is that it allows for a high degree of flexibility in the application of the ML component, however, is likely to be less efficient than a fully-embedded ML model, due to the potential for data transmission bottlenecks. This framework was then formalized by Zhong et al. (2023).

There are many examples where method 2 was used, such as Rasp et al. (2018), Brenowitz & Bretherton (2018), Gagne et al. (2019) and Gagne et al. (2020). The obvious disadvantage of this approach is that every change to the ML model being used requires reimplementation in the Fortran, and if the aim is to test a suite of ML models, this approach becomes untenable. Furthermore, this approach poses greater technical barriers for scientists developing ML-based solutions for numerical model challenges, since they must be sufficiently proficient in Fortran to reimplement models in it, rather than using existing user-friendly Python toolkits.

A solution lying somewhere between methods 2 and 3 was developed by Ott et al. (2020), who developed a Fortran-Keras Bridge (FKB) library that facilitated the implementation of Keras-like† NN modules in Fortran, providing a more modular means to build NNs in Fortran code. This however did not fully overcome the drawbacks posed by method 2 on its own; implementation of layers in the Fortran is still necessary, and any innovations in the Python modules being used would need to be mirrored in the Fortran library.

https://doi.org/10.5194/egusphere-2023-350
Preprint. Discussion started: 17 April 2023

Finally, method 3 is being tackled by the Met Office in the PIML[6] project, and by ECMWF with an application called Infero[7]. These projects both seek to develop a framework which can be used by researchers to develop ML solutions to modelling problems in Python, and then integrate them directly into the existing codebase of the physical model (e.g., the Unified model at the UK Met Office). The approach used is to directly expose the compiled code underpinning the Python modules within the physical model code.

4. Application of ML for the partial differential equations governing fluid flow

The representation and solving of the partial differential equations (PDEs) governing the fluid flow and dynamical processes in the oceans and atmosphere can be considered the backbone of weather and climate models. The solvers used to find solutions to these equations are typically iterative, and must solve the dynamics-governing equations of their model on every timestep and at every grid point. There has been growing interest in using ML to facilitate speedups and computational cost reductions in the preconditioning and execution of these solvers. Preconditioners are used to reduce the number of iterations required for a solver to converge on a solution, and usually do so by inverting parts of the linear problem. Many earlier studies focused on using ML to select the best preconditioner and/or PDE solver from a set of possible choices (e.g. Holloway & Chen, 2007; Kuefler & Chen, 2008; George et al., 2008; Peairs & Chen, 2011; Huang et al., 2016; and Yamada et al., 2018). Ackmann et al. (2020) approached the preconditioner part of the system more directly, using a variety of ML methods to directly predict the pre-condition of a linear solver, rather than using a standard preconditioner. Rizzuti et al. (2019) focused on the solver, using ML to apply corrections to a traditional iterative solver for the Helmholtz equation. Going a step further, a number of studies have used ML to replace the linear solver entirely (Ladický et al., 2015; Yang et al., 2016; Tompson et al., 2017).

Representation of the fluid equations in a gridded model poses a challenge because of the inability to resolve fine features in their solution. This leads to the use of course-grained approximations to the actual equations, which aim to accurately represent longer-wavelength dynamics while properly accounting for unresolved smaller-scale features. Bar-Sinai et al. (2019) trained a NN to optimally discretize the PDEs based on actual solutions to the known underlying equations. They showed that their method is highly accurate, allowing them to integrate in time a collection of nonlinear equations in 1 spatial dimension at resolutions 4× to 8× coarser than was possible with standard finite-difference methods.

Building on this, Kochkov et al. (2021) developed a ML-based method to accurately calculate the time evolution of solutions to nonlinear PDEs which used grids an order of magnitude coarser than is traditionally required to achieve the same degree of accuracy. They used convolutional NNs to discover discretized versions of the equations (as in Bar-Sinai et al., 2019), and applied this method selectively to the components of traditional solvers most affected by coarse resolution, with each NN being equation specific. They utilized the property that the dynamics of the PDEs were localized, combined with the convolutional layers of their NN enforcing translation invariance†, to perform their

[6] https://turbo-adventure-f9826cb3.pages.github.io/ accessed 7th February 2023
[7] https://infero.readthedocs.io/en/latest/ accessed 7th February 2023

https://doi.org/10.5194/egusphere-2023-350
Preprint. Discussion started: 17 April 2023

training simulations on small but high-resolution domains, making the training set affordable to produce. An interesting feature of their training approach, which is growing in popularity, was the inclusion of the numerical solver in the training loss function: the loss function was defined as the cumulative pointwise error between the predicted and ground truth values over the training period. In this way, the NN model could see its own outputs as inputs, ensuring an internally-consistent training process. This had the effect of improving the predictive performance of the model over longer timescales, in terms of both accuracy and stability. Finally, the authors demonstrated that their models produced generalizable properties (i.e., although the models were trained on small domains, they produced accurate simulations over larger domains with different forcing and Reynolds number). They showed that this generalization property arose from consistent physical constraints being enforced by their chosen method.

An alternative to using ML to discover discretized versions of the PDE equations is to instead use NNs to learn the evolution operator of the underlying unknown PDE, a method often referred to as a DeepONet[†]. The evolution operator maps the solution of a PDE forwards in time and completely characterizes the solution evolution of the underlying unknown PDE. Because it is operating on the PDE, it is scale invariant and so bypasses the restriction of other methods that must be trained for a specific discretization or grid scale. Interest in, and the degree of sophistication of, DeepONets has grown rapidly in recent years (e.g., Lu et al., 2019; Wu & Xiu, 2020; Bhattacharya et al., 2020; Li et al., 2020a; Li et al., 2020b; Li et al., 2020c; Nelsen & Stuart, 2021; Patel et al., 2021; Wang et al., 2021; Lanthaler et al. 2022), to the point where the method is showing promising speedups: 3x faster than traditional solvers in the case of Wang et al. (2021).

The application of ML to the solving of PDEs and the preconditioning of PDE solvers has been a fruitful avenue of research to date. It has led to innovations which have proven useful even outside of the immediate field (e.g., Pathak et al. 2022 adapted innovations from DeepONets to use in fully ML-based weather models - this is discussed further in the next Section). This is likely in part because there are many areas of engineering and science which are active in progressing relevant research, leading to a greater overall pace of innovation. ML-based PDE solvers and preconditioners have not yet been tested in a physical weather and climate model. There are few theoretical reasons this could not occur and, if effective, result in significant computational efficiencies for traditional physical model architectures. This poses an interesting avenue for further research.

5. Numerical model replacement/emulation

The shift from using ML to emulate or replace parametrization schemes to using ML to replace the entire GCM has been made plausible by the increasing volume of training data available. The focus in this section will be on the challenge of completely replacing a GCM with a ML model.

There has been a flurry of activity in the use of ML for nowcasting (e.g. Ravuri et al., 2021), however, since the focus of this review is on weather and climate applications, these studies will not be elaborated on.

https://doi.org/10.5194/egusphere-2023-350
Preprint. Discussion started: 17 April 2023

5.1. Early work – 1D deterministic models

Work on the use of ML to predict chaotic time-domain systems initially focused on 1-D problems, including 1-D Lorenz systems (e.g. Karunasinghe & Liong, 2006; Vlachas et al., 2018). Of particular interest is Vlachas et al. (2018), who used Long Short-Term Memory Networks (LSTMs†), which are well-suited to complex time domain problems. The recent popularization of convolutional LSTMs, which can also incorporate spatial information, suggests that revisiting the application of LSTMs for the prediction of spatially resolved chaotic systems could prove fruitful.

5.2. Moving to spatially extended deterministic ML-based models

Replacing a GCM entirely with an ML alternative was first suggested and tested in a spatially-resolved global configuration by Dueben and Bauer (2018), although for this study they only sought to predict a single variable (geopotential height at 500 hPa) on a 6 degree grid. Scher (2018) trained a CNN to predict the next model state of a GCM based on the complete state of the model at the previous step (i.e., an emulator of the GCM). Since this work was intended to be a proof-of-concept, the authors used a highly simplified GCM with no seasonal or diurnal cycle, no ocean, no orography, a resolution of ~625 km in the horizontal, and 10 vertical levels. Nonetheless, their ML model showed impressive capabilities; it was able to predict the complete model state several timesteps ahead, and when run in an iterative way (i.e., by feeding the model outputs back as new inputs) was able to produce a stable climate run with the same climate statistics as the GCM, with no long-term drift (even though no conservation properties were explicitly built into the CNN). Scher & Messori (2019) then extended on this, but continued the proof-of-concept approach. They investigated the ability of NNs to make skillful forecasts iteratively a day at a time to a lead time of a few days for GCMs of varying complexity, and explored a combination of other factors, including number of training years, the effects of model retuning, and the impact of a seasonal cycle on NN model accuracy and stability.

Sønderby et al. (2020) took a more targeted approach, developing a NN to produce probabilistic precipitation forecasts to a lead time of 8 hours on a 2 x 2 km resolution grid covering 7000 x 2500 km over the continental United States, with temporal resolution of 2 min and latency (execution time) in the order of seconds. The desired lead time is an input parameter and time-stepping is not used. The focus here was producing rapid high-resolution short-term forecasts of a single key variable. Weyn et al. (2019) also aimed to predict a limited number of variables but focused more on the NWP to medium range time domain. They trained a CNN to predict 500 hPa geopotential height and 300 to 700 hPa geopotential thickness over the Northern Hemisphere to up to 14-days lead time, showing better skill out to 3 days than persistence, climatology, and a dynamics-based barotropic vorticity model, but not better than an operational full-physics weather prediction model.

Weyn et al. (2020) then improved on this significantly, with a Deep U-Net style CNN trained to predict four variables (geopotential height at 500 and 1000 hPa, 300 to 700 hPa geopotential thickness, and 2 m temperature) globally to 14 days lead time. A major innovation in this study was their use of a cubed-sphere grid, which minimized distortions for planar convolution algorithms while also providing closed boundary conditions for the edges of the cube faces. Additionally, they extended their previous work to include sequence prediction techniques, making skillful predictions

https://doi.org/10.5194/egusphere-2023-350
Preprint. Discussion started: 17 April 2023

possible to longer lead times. Their improved model outperformed persistence and a coarse resolution comparator (a T42 spectral resolution version of the ECMWF IFS model, with 62 vertical levels and ~2.8 degree horizontal resolution) to the full 14 days lead time, but was not as skillful as a higher resolution comparator (a T63 spectral resolution version of the IFS model with 137 vertical levels and ~1.9 degree horizontal resolution) or the operational subseasonal-to-seasonal (S2S) version of the ECMWF IFS.

Inching slightly closer to being competitive with physical models, Rasp & Thuerey (2021) developed a ResNet DNN model trained to predict geopotential height, temperature and precipitation to 5 days lead time and assessed it against the same set of physical models as Weyn et al. (2020). Their model was close to as skillful as the T63 spectral resolution version of the IFS model, and had better skill to the 5 day lead time than Weyn et al. (2020).

Keisler (2022) took an ambitious step forward, training a Graph Neural Network† (GNN) model to predict 6 physical variables on 13 atmospheric levels on a 1-degree horizontal grid, which the authors claim is ~50-2000 times larger than the number of physical quantities predicted by the models in Rasp & Thuerey (2021) and Weyn et al. (2020). Their model worked by iteratively predicting the state of the 6 variables 6 hours into the future (i.e., the output of each model timestep was the input into the next timestep), to a total lead time of 6 days. The authors showed that their model outperformed both Rasp & Thuerey (2021) and Weyn et al. (2020) in the variables common to all three studies. They suggested that the gain in skill seen over previous studies was due to the use of more channels† of information, and the higher spatial and temporal resolution of their model. Finally, they showed that their model was more skillful than NOAA's GFS physical model to 6 days lead time, but not as skillful as ECMWF's IFS.

Lam et al. (2022) also used GNNs to build their ML-based weather and climate model, GraphCast. This model was the most skillful ML-based weather and climate model at the time of writing this review. While the first ML-based weather and climate model to claim to exceed the skill of a numerical model was Pangu-Weather (Bi et al., 2022; described in greater detail in the following subsection), GraphCast exceeded the skill of both the ECMWF deterministic operational forecasting system, HRES, and also Pangu-Weather. Furthermore, Lam et al. (2022) paid particular attention to evaluating their model and HRES against appropriate measures, and included existing model assessment scorecards from ECMWF to evaluate them. GraphCast capitalized on the ability of GNNs to model arbitrary sparse interactions by adopting a high-resolution multi-scale mesh representation of the input and output parameters. It was trained on the ECMWF ERA5 reanalysis archive to produce predictions of five surface variables and six atmospheric variables, each at 37 vertical pressure levels, on a 0.25° grid. It made predictions on a 6-hourly timestep and was run autoregressively to produce predictions to a 10-day lead time. The authors demonstrated that GraphCast was more accurate than HRES on 90.0% of the 2760 variable and lead time combinations they evaluated.

5.3. Probabilistic ML-based models

A common criticism of ML approaches to weather and climate prediction is the difficulty of representing uncertainty and extremes. For example, Watson (2022) argued that while there is now an abundance of examples of ML being used for model parameterization schemes, full model replacement, downscaling, and PDE solvers (much of which is

https://doi.org/10.5194/egusphere-2023-350
Preprint. Discussion started: 17 April 2023

covered in this review), there are relatively few examples which address the question of how well ML approaches can reproduce extreme events and statistics. There are however a growing number of examples where this is now being considered, many of which fall into the category of full-model replacement.

Clare et al. (2021) tackled this challenge by training a NN to predict full probability density functions of geopotential height at 500 hPa and temperature at 850 hPa at 3 and 5 days lead time, producing a probabilistic forecast which was comparable in accuracy to Weyn et al. (2020).

Weyn et al. (2021) have also explored probabilistic ML predictions using an ensemble of DNNs similar to the one described in Weyn et al. (2020). The authors expanded the number of variables predicted from 4 to 6, and used initial condition perturbation methods and variations in atmospheric representation similar to those used in traditional ensemble prediction to generate the ensemble of DNN predictions. They generated 320-member ensembles (much larger than could be affordably achieved with a physics-based model) and produced forecasts to 6 weeks lead time - considerably longer than any comparable work to date. The skill of the ensemble mean of the system, a control member, and the full ensemble were assessed against the same metrics from the ECMWF sub-seasonal to seasonal (S2S) prediction system. Their ML ensemble model had lower skill than the S2S system at shorter lead times, but was comparable in skill at longer lead times.

Pathak et al. (2022) developed a weather model called FourCastNet, leveraging the work on DeepONets described in Section 4. In particular, the authors used a type of DeepONet called a Fourier Neural Operator (FNO). FourCastNet produced predictions of 20 variables (including challenging-to-predict variables such as surface winds and precipitation) on five vertical levels with 0.25 degree horizontal resolution, and had competitive skill against the ECMWF IFS to 1 week lead time. The high horizontal resolution of their model enabled it to resolve extreme events such as tropical cyclones and atmospheric rivers, and the speed of the model facilitated the generation of large ensembles (1,000's of members). This suggests that prediction of very extreme events may be possible. The authors make the ambitious claim that with additional resources and further development, they anticipate that FourCastNet could match the capabilities of current NWP models on all timescales and at all vertical levels of the atmosphere.

Bi et al. (2022) achieved a significant milestone with their model Pangu-Weather, the first ML-based model to perform better than the ECMWF IFS to a lead time of 7 days based on RMSE and Anomaly Correlation Coefficient (ACC) across several variables including geopotential height and temperature at 500 hPa. Pangu-Weather featured two major innovations over FourCastNet:

1. It used 3D (latitude, longitude and height) input grids trained against 3D output grids. This enabled different levels of the atmosphere to share information, which was not possible in FourCastNet, in spite of predicting variables on multiple atmospheric levels, because the levels were treated independently. In contrast, Pangu-weather adopted a 3D convolutional method the authors name the 3D Earth-specific transformer (3DEST), which enabled the flow of information both horizontally and vertically.
2. It was made up of a series of models trained with different prediction time gaps. The motivation for this was that, as noted by the authors, when the goal is to produce forecasts to 5 days (for example), but the timestep

https://doi.org/10.5194/egusphere-2023-350
Preprint. Discussion started: 17 April 2023

of the basic forecast model is relatively short (e.g. 6 hours), many iterative executions of the model are required, with the errors of each iteration feeding onto the next. A shorter model timestep results in greater overall errors (due to more iterations being required to reach the final forecast lead time), and a longer model timestep reduces this error. Motivated by this, the authors trained several versions of their model to predict to different timesteps on a single iteration. The overall forecast to a given leadtime was then constructed using the longest possible timesteps. For example, for a 7-day forecast, a 24-hour forecast is iterated 7 times, whereas for a 23-hour forecast, a 6-hour forecast is iterated 3 times, followed by a 3-hour forecast 1 time, and 1-hour forecast 2 times. The authors noted that this strategy was not effective to multiweek or longer timescales; they reported that training the model with a 28-day timestep was difficult for example, and suggested that more powerful or complex ML methods would be required to achieve this.

As well as the relatively broad measures of RMSE and ACC, the authors assessed the ability of their system to represent the intensity and track of selected tropical cyclones, and explored the potential for producing useful ensemble forecasts. They found that Pangu-Weather predicted the tracks of the cyclones considered with a high degree of accuracy compared to the ECMWF IFS, however it underestimated cyclone intensity. The authors attributed this to the training data they used (ERA5) also underestimating cyclone intensity. To assess ensemble predictions, they perturbed the initial state of the system with Perlin noise vectors to produce a 100-member ensemble of forecasts and calculated the RMSE and ACC of the ensemble mean for selected variables. They note that the ensemble mean forecasts performs worse than a single deterministic forecast for shorter lead times (e.g., 1 day), but better for longer lead times. The authors did not, however, investigate the properties of the spread of the ensemble or its utility for probabilistic forecasts or predicting statistical extremes.

As already mentioned above, the skill of Pangu-Weather was exceeded by GraphCast, although Lam et al. (2022) only assessed GraphCast in deterministic setting. Nonetheless, there is nothing stopping GraphCast from being used in an ensemble mode, and the assessment of GraphCast presented by Lam et al. (2022) was much more comprehensive and exacting than the assessment of Pangu-Weather presented by Bi et al. (2022).

It should be noted that all of the major milestones and high-profile ML models described in this section have relied on reanalysis datasets produced by physics-based models. The provision of higher resolution and higher quality open datasets have the potential to drive progress in this area as much as, if not more than, improvements and further research into ML algorithms.

5.4. Moving to more extensible models

As the effectiveness of ML approaches are increasingly demonstrated in the literature, additional factors become clear in considering these models for both research and application. In a research setting, the ability to readily perform transfer learning to new problems and reduce training costs will be significant in supporting adoption by other researchers.

https://doi.org/10.5194/egusphere-2023-350
Preprint. Discussion started: 17 April 2023

This need for greater flexibility in both the input data sources and predictive outputs of ML weather and climate models was recognized by Nguyen et al. (2023), who developed a transformer architecture-based ML model called ClimaX. This model was designed as a foundational model, trained initially on datasets derived from the CMIP6 (Eyring et al., 2016) dataset, and designed to be readily retrained to specific tasks using transfer learning. The authors demonstrated the skill of ClimaX against simpler ML models, and in some cases a numerical model (ECMWF IFS), for a variety of tasks including weather prediction, sub-seasonal prediction, climate scenario prediction, and climate downscaling. The authors showed that ClimaX was able to make skillful predictions in scenarios unseen during the initial CMIP6 training phase. Furthermore, ClimaX used novel encoding and aggregation blocks in its architecture to achieve much more affordable training compute costs than other ML weather and climate models such as Graphcast, Pangu-weather and FourCastNet.

5.5. Benchmark datasets for ML weather models

Providing open benchmark data for machine learning challenges has been as transformational for the machine learning field as improved algorithms, the publication of papers, or improvements in hardware.

As the interest and activity in the use of ML as a potential alternative to knowledge-based numerical GCMs has grown, the need for consistent benchmarks for the intercomparison of ML-based models has become increasingly clear. Rasp et al. (2020) addressed this need with the introduction of WeatherBench. On this platform, the authors provided data derived from the ERA5 archive that has been simplified and streamlined for common ML use cases and use by a broad audience. They also proposed a set of evaluation metrics which facilitate direct comparison between different ML approaches, and provided baseline scores in these metrics for simple techniques such as linear regression, some deep learning models and some GCMs. Weyn et al. (2020) chose datasets and assessment metrics consistent with WeatherBench to facilitate intercomparison of results. Rasp & Thuerey (2021) directly used the benchmarks provided by WeatherBench in their assessment. They demonstrated that their model outperformed previous submissions to WeatherBench, highlighting its value as a tool to allow intercomparability of ML-based weather models. Other examples of studies using WeatherBench data and analysis methods are Clare et al. (2021) and Weyn et al. (2021). The parameters of a good benchmark dataset were further elucidated by Dueben et al. (2022), who provided an overview of the current status of benchmark datasets for ML in weather and climate in use in the research community and provided a set of guidelines for how researchers could build their own benchmark datasets.

At the time of writing this review, assessments of ML-based models had chiefly (but not exclusively) focused on simple statistics like globally-averaged RMSE, and not reported in detail on the degree to which they accurately captured specific processes such as cyclone formation, climate drivers such as the El Nino Southern Oscillation, or large scale structures such as the jetstreams. A useful contribution from the scientific community would be to better quantify and articulate a suite of tests and statistics that could form a 'report card' to provide better insight into the value of new ML models.

https://doi.org/10.5194/egusphere-2023-350
Preprint. Discussion started: 17 April 2023

5.6. A hybrid approach

Arcomano et al. (2022) present an approach which straddles the theme of this section and that of the following section (physics-constrained ML models). Following Wikner et al. (2020), they used a numerical atmospheric GCM and a computationally-efficient ML method called reservoir computing in a hybrid configuration called Combined Hybrid-Parallel Prediction (CHyPP). Their hybrid model is more accurate than the GCM alone for most state variables to a lead time of 7-8 days. They also demonstrate the utility of their hybrid model for climate predictions with a 10-year long climate simulation, for which they showed that the hybrid model had smaller systematic errors and more realistic variability than the GCM alone.

5.7. ML for predicting ocean variables

More recently, greater attention has been paid to the application of ML to the ocean, particularly for seasonal to multi-year prediction. Initial work in this space focused on directly predicting key indices such as the NINO 3.4 index. For example, Ham et al. (2019) trained a CNN to produce skillful El Niño Southern Oscillation (ENSO) forecasts with a lead time of up to one and a half years. A limiting factor for the application of ML to ocean variables is the lack of availability of observational data for training. To overcome this, the authors used transfer learning† to train their model first on historical simulations, and then on a reanalysis from 1871 to 1973. Data from 1984 to 2017 was reserved for validation. Ham et al. (2021) improved on this by including information about the current season in the network inputs as one-hot vectors†. Including this seasonality information led to an overall increase in skill relative to the model in Ham et al. (2019), in particular for forecasts initiated in boreal spring, a season which is particularly difficult to predict beyond.

Kim et al. (2022) improved on the performance of the 2D CNNs used in Ham et al. (2019) and Ham et al. (2021) for predicting ENSO by instead using a convolutional LSTM network with a global receptive field †. The move to a larger (global) receptive field for the convolutional layers enabled the network to learn the large-scale drivers and precursors of ENSO variability, and the use of a recurrent† architecture (in this case LSTM) facilitated the encoding of long-term sequential features with visual attention†. This led to a 5.8% improvement of the correlation coefficient for Nino3.4 index prediction and 13% improvement in corresponding temporal classification with a 12-month lead time compared to a 2D CNN.

Taylor & Feng (2022) moved from prediction of indices to spatial outputs, training a Unet-LSTM† model on ECMWF ERA5 monthly mean Sea Surface Temperature (SST) and 2-m air temperature data from 1950-2021 to predict global 2D SSTs up to a 24-month lead time. The authors found that their model was skillful in predicting the 2019-2020 El Niño and the 2016-2017 and 2017-2018 La Niñas, but not for the 2015-2016 extreme El Niño. Since they did not include any subsurface information in their training data (in contrast to Ham et al. (2019) and Ham et al. (2021), who included ocean heat content), they concluded that subsurface information may have been relevant for the evolution of that event.

https://doi.org/10.5194/egusphere-2023-350
Preprint. Discussion started: 17 April 2023

It is clear from the small number of (but rapidly evolving) studies in this space that there is great promise for the use of ML for seasonal and multi-year prediction of ocean variables, with many avenues to pursue to achieve potential skill gains.

5.8. ML for climate prediction

The literature on the use of ML for prediction on seasonal to climate timescales is still relatively sparse compared to its use for nowcasting and weather prediction. Some examples have been covered in previous sections, such as Weyn et al. (2021) on subseasonal to seasonal timescales in the atmosphere, and Ham et al. (2019), Ham et al. (2021), Kim et al. (2022) and Taylor & Feng (2022) on seasonal to multiyear timescales in the ocean. A major cause for this sparsity is that deep learning typically requires large training datasets, and the available observation period for the earth system is too short to provide appropriate training data for seasonal to climate timescales in most applications. In the subseasonal to seasonal end, this may be overcome by including more slowly-varying fields in the training (e.g. ocean variables), by designing models to learn the underlying dynamics which drive long-term variability, and by including more physical constraints on the models. On the climate end these same methods could be beneficial, as well as transfer learning, as is done in Ham et al. (2019), and data augmentation† techniques. Additionally, interest is increasing in the use of ML to predict weather regimes and large-scale circulation patterns, which may prove beneficial in informing seasonal and climate predictions (Nielsen et al., 2022).

With the growing maturity of the field of ML for weather and climate prediction, there is every reason to believe the challenges of prediction on seasonal to climate timescales can be overcome.

6. Physics constrained ML models

As has been briefly touched on in previous sections, a promising and increasingly popular method for improving the performance of ML applications in weather and climate modelling is to include physics-based constraints in the ML model design (e.g. Karpatne et al., 2017; de Bézenac et al., 2017; Beucler et al., 2019; Yuval et al., 2021; Beucler et al., 2021; Harder et al., 2022). This can be done through the overall design and formulation of the model, and through the use of custom loss functions which impose physically-motivated conservations and constraints.

An excellent review of the possible methods for incorporating physics constraints into ML models for weather and climate modelling, along with 10 case studies of noteworthy applications of these methods, is presented in Kashinath et al. (2021). The scope of Kashinath et al. (2021) is broad and includes studies not applied directly in the context of weather and climate modelling, but applicable to it. Rather than repeat the total of this summary here, the reader is directed to this review.

A class of physics-leveraged ML which has grown rapidly in popularity is Physics Informed Neural Networks (PINNs). These are discussed in Kashinath et al. (2021), but have also become a very active area of research since the publication of that review. A more up-to-date review of this class of NNs is presented by Cuomo et al. (2022), along with a review of other related Physics guided ML architectures.

https://doi.org/10.5194/egusphere-2023-350
Preprint. Discussion started: 17 April 2023

While PINNs are an exciting and promising new NN architecture, they still face some challenges. For example, they have had little success simulating dynamical systems whose solution exhibits multi-scale, chaotic or turbulent behavior. Wang et al. (2022b) attributed this to the inability of PINNs to represent physical causality, and developed a solution by re-formulating the loss function of a PINN to explicitly account for physical causality during model training. They demonstrated that this modified PINN was able to successfully simulate chaotic systems such as a Lorenz system, and the Navier-Stokes equations in the turbulent regime; something which traditional PINNs were unable to do.

Nonetheless, recent work with PINNs has led to some interesting results for weather and climate simulation: Bihlo & Popovych (2022) used PINNs to solve the shallow-water equations on a rotating sphere, as a demonstration of their utility in a meteorological context, and Fuhg et al. (2022) developed a modified PINN to solve interval and fuzzy partial differential equations, enabling the solving of PDEs including uncertain parameter fields.

7. Other applications of ML and considerations for the use of ML in Weather and Climate Models

Aside from the most active areas of development in the use of ML in weather and climate models discussed in the sections above, there are a few areas of the literature worth mentioning that are adjacent to the main focus of this review. These topics are covered in the following subsections.

7.1. Nudging

Rather than replacing a component or components of a GCM with an ML alternative to gain skill improvements, Watt-Meyer et al. (2021) focused on using corrective nudging to reduce model biases and the errors they can introduce through feedbacks. The authors used RFs to learn bias-correcting tendencies from a hindcast nudged towards observations. They then coupled this RF to a prognostic simulation and attempted to correct the model drift with the learned nudging tendencies. While this simulation ran stably over the year-long test period and showed improvements in some variables, the errors in others were observed to increase. So far studies in this space seem to be limited to Watt-Meyer et al. (2021), however this method seems promising, so hopefully interest in developing this approach further will grow in the future.

7.2. Object identification within models

An alternative to achieving greater model accuracy through increasing resolution of the entire model grid is to develop techniques to identify critical systems and physical phenomena within the model, and embed higher resolution temporary subgrids within the larger GCM to more accurately simulate those processes. A key challenge to overcome to achieve this is automatically identifying key model features. For example, Mudigonda et al. (2017) investigated the feasibility of using a variety of NN architectures to identify storms, tropical cyclones and atmospheric rivers within model data, with promising results. A major limitation of this area of research is the frequent need for labelled datasets of the events being identified, which are currently quite limited. While there are approaches to this problem which

https://doi.org/10.5194/egusphere-2023-350
Preprint. Discussion started: 17 April 2023

utilize unsupervised learning (i.e., learning without an objective function or labelled data), it is harder to achieve a meaningful result this way.

7.3. Uncertainty quantification

A common criticism of some ML models such as NNs is that it is difficult to represent the uncertainty of their outputs. Some examples of studies that have sought to overcome this have already been mentioned in Section 3.8, and there are other examples in the literature (e.g. Grigo & Koutsourelakis, 2019; Atkinson, 2020; Yeo et al., 2021; O'Leary et al., 2022), however it is nonetheless still a relatively underexplored aspect of ML models for physical systems. Psaros et al. (2022) suggest that this may be because they are also under-utilized within the broader deep learning community, and it is thus a developing field that is not universally trusted and understood yet. They also point out that the physical considerations inherent to ML applied to physical systems often make them more complicated and computationally expensive than standard ML applications, further disincentivizing the inclusion of uncertainty quantification in an already complex problem.

Only recently has attention to this aspect of ML become sufficient to motivate the collection of methods into a consistent framework, a good example of which is the aforementioned Psaros et al. (2022), who presented a comprehensive review of the methods for quantifying uncertainty in NNs and provided a framework for applying these methods.

A related topic which is facing similar challenges is the question of explainability of ML approaches; often there is value in understanding the relative roles and importance of predictors in an ML model, or the relative significance of different regions of the predictor data. Flora et al. (2022) provide a good overview of approaches to this and compare their relative drawbacks and benefits.

7.4. GPUs and specialized compute resources

GPUs and TPUs are specialized hardware which are well suited to highly parallelizable matrix operations, ideal for solving neural network operations. TPUs have been developed specifically for deep learning applications. Both GPUs and TPUs are likely to be available on many of the next generation of supercomputers, but much of the current Fortran-based numerical weather and climate model infrastructure cannot be run on them in their current state. Data bottlenecks also exist between the GPUs (which have their own on-board memory) and the main memory accessible to the CPU. While efforts are underway to make numerical and climate models better suited to GPUs, for example with the development of LFRic (Adams et al. 2019), the new weather and climate modelling system being developed by the UK Met Office to replace the existing Unified Model (Walters et al. 2017), there is still a long way to go before entire weather and climate models can be reliably run on GPU or other specialized compute architectures. At the same time, some neural network designs are aimed squarely at the partial differential equation solving at the core of numerical methods. Since neural network evaluation utilizes simpler mathematical operations than current PDE solvers, they offer the prospect of significant computational advantages on non-specialized (i.e., CPU) hardware.

https://doi.org/10.5194/egusphere-2023-350
Preprint. Discussion started: 17 April 2023

8. Key papers from computer science

This section provides a brief perspective on weather and climate modelling from the computer science domain, and aims to provide the earth system scientist with a short list of the main relevant innovations in computer science.

As was noted in Section 1, ML models are often regarded as black-boxes, largely because of the design of many prominent ML systems. In principle, it is not quite right to refer to a model as "a machine learning model", in the sense that either a physical model or a NN could undergo a training cycle to determine optimal parameter values. The parameters of a neural network are its weights and biases, whereas the parameters of a physical model are physical variables and constants. The essence of ML is the level of automation involved. Even in typical ML models such as large NNs, the model architecture is typically specified manually by the data scientist or physical scientist involved. The automated derivation of model architecture and composition is not yet mature for large models, although it is explored through evolutionary programming techniques whereby the learning of architecture as well as parameterization is automated.

As such, the goal of a "ML weather/climate model" (either a full model or an augmented numerical model) will likely be achieved using multiple model types and architectures, in a complex fashion.

Some models from the computer science domain also have a more fundamental probabilistic or statistical underpinning than typical weather and climate models in that the model state variables comprise a probability distribution or degree of confidence. Whereas a typical weather or climate model derives its probability outputs from an ensemble of perturbed members, an alternative approach could be taken whereby each part of the belief state[†] of the model is a distribution or likelihood, built up either empirically or by fitting a gaussian or other known distribution.

As such, ML may be applied to statistical models, process-based models, Bayesian models or physical models. Nonetheless, current directions in ML are focused on very large or deep NNs which rely both on the universal approximation theorem and practical experimentation to capture a prediction function without needing to explicitly represent the processes being modeled. In a conceptually similar fashion to how a Fourier decomposition can represent any wavelike function, the universal approximation theorem establishes that a NN may approximate any function, subject to its size and the required degree of accuracy (Hornik, Stinchcome and White 1989). Deep learning has been highly effective in approaching many problems, but many limitations are acknowledged, as evidenced by the current widespread focus on trustworthy computing and efforts towards explainable ML systems.

8.1. A Selected Timeline of Machine Learning:

In this section a selected history of ML developments relevant to weather and climate modelling is presented. The purpose of this is to give the reader some perspective on the timelines of the innovations in this space, and provide some additional context for the current state of the field.

- 1943: First development of a mathematical model of a human neuron by McCulloch & Pitts (1943)
- 1950: The Turing Test (originally called The Imitation Game) was developed (Turing, 1950)

https://doi.org/10.5194/egusphere-2023-350
Preprint. Discussion started: 17 April 2023

- 1956: Arthur Samuel developed the first automated checkers program (one of the earliest examples of using machine learning to exceed the human skill of the developer)
- 1974: Development of backprogatation as a sound way of updating a parameterization based on model errors (Werbos, 1974, 1990). This work showed how to apply machine learning to artificial neuron models (i.e., NNs).
- 1989: Hornik et al. (1989) proved that NNs can effectively approximate any function. This had theoretical implications for speedups to equation solvers which could theoretically be solved with orders of magnitude greater efficiency
- 1996: The LSTM architecture was developed, addressing the vanishing gradient problem (Hochreiter & Schmidhuber, 1996). LSTMs and their successors would revolutionize language modelling, speech recognition, translation, and text-to-speech synthesis. They can be used in any tokenized sequence modelling and have many applications in time-series prediction.
- 1998: The release of the Modified National Institute of Standards and Technology (MNIST) database spurred a generation of machine learning research into novel methods, coinciding with and stimulating major research and application of new techniques including DTs and NN approaches (LeCun et al., 1998). This milestone demonstrates how the release of open data, in an accessible way which supports machine learning patterns, can accelerate progress in a research field. Many people's first experience with machine learning is to train a model for MNIST prediction.
- 2001: Breiman (2001) developed RFs. This CPU-efficient algorithm found widespread application, and evolutions of this approach remain relevant today. This approach was extended to gradient-boosting by Freidman (2001).
- 2009: ImageNet data set released (Deng et al., 2009). Called "the data that transformed AI research", this milestone shows again how a key dataset release designed to support machine learning research can accelerate an entire field. This was also an important moment in the AI industry better recognizing the importance of data quality as much as algorithm complexity to AI performance.
- 2014: Goodfellow et al., (2020) introduced GANs (note that while this paper was ultimately published in 2020, a version of the paper was available on ArXiv in 2014[8]). GANs introduce the concept of a generator[†] model, and a discriminator model which aims to distinguish predictions from observations. While the generator learns the predictive function, its objective function is the discriminator. When the discriminator can no longer distinguish the prediction from an observation, the overall model has been optimized. This has applications in producing highly realistic gridded outputs.

[8] https://doi.org/10.48550/arXiv.1406.2661, accessed 7th February 2023

https://doi.org/10.5194/egusphere-2023-350
Preprint. Discussion started: 17 April 2023

- 2015: Ronneberger et al. (2015) introduced U-Nets, which are useful for image segmentation. Image segmentation has application in locating weather features such as fronts, cyclones, convection etc.
- 2015: He at al. (2016) introduced CNNs with skip-connections to improve performance and allow attention-like mechanisms.
- 2017: Vaswani et al. (2017) provided a significant step forward for sequence transduction modelling (i.e., language translation) using an 'attention' mechanism which allows models to consider contextual factors in sequence prediction. Attention mechanisms let a model draw from the state at any preceding point along the sequence, allowing long sequences to be learned. This approach is replacing recurrent architectures for many applications.
- 2020: Dosovitskiy et al. (2021) applied and extended attention-based techniques to image processing with reduced computational requirements.

This history shows the degree and rate of research into processing images, text and other sequences based on semantic understanding of content, but does not demonstrate capturing physical processes as a core element. Advances in the weather and climate modelling domain have a more explicit goal of properly portraying real physical processes. Bringing these concepts together promises to uplift capability in both fields.

9. Practical Perspectives on Machine Learning for Weather and Climate Models

A major driver of research into, and improvement of, weather and climate models is increasing the skill of operational forecast systems around the world, and increasing the accuracy and trustworthiness of climate projections. Therefore, an important consideration for ML in the context of weather and climate models is the need for it to ultimately be integrated into a complete predictive system with practical application for forecasting or climate projections.

The research findings covered in this review, however compelling, are yet to make major changes to operational modelling systems, or standard climate projections. There are a number of challenges (which are discussed further in Section 10) that need to be overcome to leverage and capitalize on the new capabilities which have been developed in research settings.

This section summarizes some practical considerations the research community may wish to be aware of.

A major function of operational meteorological services is to inform of future conditions, largely for managing risk or optimizing benefits. A conservative approach is taken, which is to say, the utmost premium is put on accuracy, resilience, reliability, and solid scientific foundation. There is often a large gap between a research finding and the ability to successfully integrate an innovative new approach into a major model upgrade or scientific configuration upgrade. Understanding when to invest effort in bringing a research innovation into a major model or scientific configuration upgrade is a significant challenge requiring a great deal of time and effort. This is also true of the transition into operations for operational agencies.

https://doi.org/10.5194/egusphere-2023-350
Preprint. Discussion started: 17 April 2023

One pathway to adoption of weather and climate models that use ML could be the development of limited-scope models optimized for one or a few parameters. Early effectiveness of limited-purpose ML models provide the ability to augment existing services without disruption. A risk associated with this approach is that inconsistencies between predictions may arise from independent forecasts from a collection of limited-scope models, leading to confusion from users and an erosion of trust.

Such an approach could be considered to entirely replace current model systems at the cost of consistency and model fidelity. Operational development is typically more incremental, however. It is more likely that progress will be made in achievable increments, along the evolving technical frontier. However promising and fascinating as a research direction, full model replacement is currently not mature enough for an operational system.

What may currently be operationally feasible includes parameterization scheme replacement, solver replacement, super-resolution, new approaches to data assimilation of novel observation sources, and both pre- and post-processing applications (although of course not all these applications have been covered in this review).

It is expected that the research into, and application of, ML methods will represent an increasing proportion of weather and climate model research, with increasingly sophisticated and skillful model components finding their way into major model releases over the coming years. These components are appealing for both computational and model skill reasons, and are expected to be highly promising avenues of research.

10. Conclusions

In this review we have presented a comprehensive survey of the literature on the use of ML in weather and climate modelling.

We have found that the ML models being most often explored include RFs, and NNs and DNNs (including ResNets and Deep U-Nets). We have also identified some recent innovations which have proven to be highly effective in the weather and climate modelling space, including DeepONets and variants thereof, and PINNs.

This review has demonstrated that ML is being successfully applied to many aspects of weather and climate modelling. We have presented examples from the literature of its application in (1) the emulation and replacement of sub-grid scale parametrizations and super-parametrizations, (2) preconditioning and solving of resolved equations, (3) full model replacement, and (4) a selection of other adjacent areas.

Nonetheless, there are still many challenges to overcome, including:

- addressing the instabilities excited in physical models due to the inclusion of ML components;
- increasing the ease of technical integration (in particular, Fortran compatibility);
- memory and computational concerns;

https://doi.org/10.5194/egusphere-2023-350
Preprint. Discussion started: 17 April 2023

- representing a sufficient number of physical parameters and increasing physical and temporal resolution in ML-based weather and climate model implementations (which currently feature reduced fields and levels compared to physics-based numerical models);
- moving from a focus on individual parts of the earth system (i.e., the atmosphere, the ocean, the land surface etc.) to tackling the challenges associated with coupled models (i.e., where models of individual components of the earth system are coupled together). Increasingly, operational weather and climate models are coupled land-atmosphere-ocean-sea-ice models in order to more accurately represent the relevant timescales and processes in the earth system, and ML modelling efforts need to reflect this;
- the need for more good quality training data; and
- the practical challenges of integrating ML components or models into an operational setting.

This list provides a set of focus areas for future research efforts.

If the current trend in skill gains in full ML weather and climate models continues, it is possible they will eventually be considered viable alternatives to traditional numerical models. However, a more likely scenario is that ML components will replace an increasing number of physics-based model components, with models the near-term future being hybrid ML-physical models.

Some possible avenues through which increases in ML-based weather and climate model skill might be achieved is by operating at higher resolutions, resolving more processes which are implicit in the training data, or by undertaking experiments on synthetic data to address the paucity of real-world data.

Another benefit of ML approaches to weather and climate modeling is the relative computational cheapness of ML alternatives to current physics-based modelling systems. This has the potential to open the door to experiments that would not be feasible otherwise. For example, experiments requiring a very large ensemble would be more feasible with a computationally cheap ML approach.

The literature reviewed here indicates that 'out of the box' ML approaches and architectures are not effective when used in a weather and climate modelling context. Rather, ML architectures must be adapted to satisfy conservation of energy, represent physically realistic predictions and processes, and maintain good model stability. At the same time, computational and memory tractability must be maintained.

Advances in the sophistication, complexity and efficiency of ML architectures are being heavily invested in for many use cases in other disciplines and in the private sector (e.g., condition-action post estimation, text to video generation, stable diffusion/text to image, chatbots, facial recognition, semantic image decomposition, etc.). In order to capture the full benefits of ML for the weather and climate modelling domain, academic and operational agencies will need to continue to support research in this space.

https://doi.org/10.5194/egusphere-2023-350
Preprint. Discussion started: 17 April 2023

Interest and progress in the application of ML to weather and climate modelling has been present for close to 30 years, and has begun to accelerate rapidly in the last few years. There is good reason to believe that ML as a tool will have transformational (and potentially highly fascinating, innovative, and offer great opportunity for further application.

Machine Learning Glossary of Terms

This glossary includes terms which the reader will come across frequently in machine learning literature for the weather and climate, as well as in machine learning literature generally. Most of these terms are used in this paper while others support further reading.

Activation function. The function which is used to multiply input values, add the bias and produce an output value from an individual node. Examples include linear, sigmoid and tanh.

Adversarial attack. The deliberate use of malicious data input in a real-world setting intended to cause a misclassification, underperformance or unexpected behaviours. Examples include emails designed to avoid spam filters, or images that have been modified to avoid recognition.

Adversarial example. A specialised input which results in a misclassification or underperformance of a predictive model. An example of this concept is an image which has had subtle noise added to it resulting in a copy of that image which is visually indistinguishable from the original, but which nonetheless causes a misclassification. The term 'adversarial' is used to refer to the way the example fools the model and is not necessarily intended to convey the sense of malicious intent, although the term is often applied in that fashion. Adversarial examples demonstrate that machine learning models may be more brittle than expected based on ordinary training data alone. To increase model robustness, adversarial examples may be generated and added to the training set. Data augmentation techniques such as flipping, warping and adding noise (any many other techniques) are also used to generate additional training data to increase robustness and performance.

Attention mechanism. A complex mechanism to allow sequence prediction models to increase the importance of key terms within that sequence which may be nonlocal and modified in meaning according to the other terms of the sequence.

API. Application Programming Interface. A set of programming functions, methods or protocols by which to build and integrate applications. APIs may be "web" APIs or imported from software packages in which case they are more often referred to as libraries.

Autoencoder. A neural network architecture which learns to produce a 'code' for an input sequence from which the original data can be retrieved. The code is shorter than the original input sequence. Applications include data compression and denoising data.

Back propagation. A process of utilising the errors from a prediction to update the weights and biases of a neural network.

Batch. See training batch.

https://doi.org/10.5194/egusphere-2023-350
Preprint. Discussion started: 17 April 2023

Batch normalisation. Data normalisation which aligns the means and variances of input data to a model. For computational reasons, this is performed separately for each training batch.

Belief state. The current state of the world which is believed to be true according to a model. A common architecture in realtime applications whereby a belief state is updated according to an update function on the basis of new observations.

Channel. An additional dimension to data which is usually not a spatial dimension. Examples include the red, green and blue intensity images which comprise a colour image. Another example could be to represent both temperature and wind speed as channels.

Classification. A model which attempts to diagnose or predict the category, label, class or type that an example falls within.

Climatology. Refers to the usual past conditions for a location at a time of year. Usually calculated by temporal mean across years of a dataset, for a given time interval within those years (e.g., for a dataset of monthly mean values spanning all months of all years from 1990 to 2020, the monthly mean climatology would be obtained by averaging across all the Januarys from each year, all the Februarys, etc., to obtain an "average January", an "average February", etc.). Climatologies are often used in the same manner as persistence as a baseline prediction against which to measure a predictive model. For example, a model predicting a value for January could be compared to the climatological monthly mean value for January. This helps answer the question "is my model a better source of information than using the average past conditions from this time of year?".

Connectome. The connections between nodes in a neural network. Examples include fully-connected, partially-connected, skip-layer connections, recurrent connections and others. The 'wiring diagram' for the network.

Convolutional neural network. A deep neural network architecture commonly applied to images which utilises a small convolutional (spatially connected) kernel applied in a sliding window fashion.

Data augmentation. The practice of modifying input data in supervised learning to produce additional examples. This can make networks more robust to new inputs and address issues of brittleness to adversarial examples. An example of data augmentation is using rotated or reflected versions of the same image as independent training samples.

Data driven. A generalised term used to indicate a primary reliance or dependence on the collection or analysis of data. Used in contrast to process driven or theory driven.

Decision tree. A tree-like, or flowchart-like, branching model representing a series of decisions and their possible consequences. Each internal node represents a 'test' (i.e. decision threshold) and each leaf node represents a class label or collection of possible outcomes.

Deep neural network. A neural network with many layers. Deeper, thinner networks have proven easier to train than wider, shallower ones.

https://doi.org/10.5194/egusphere-2023-350
Preprint. Discussion started: 17 April 2023

DeepONet. A deep neural network architecture relying on universal approximation theorem to train a neural network to represent a mathematical operation (the operator), such as a partial differential equation or dynamic system.

Discriminator model. A model which distinguishes or discriminates between synthetic data and real-world observations. Often used in conjunction with a generator. In this case, the overall goal is to produce a generator which is capable of fooling the discriminator, producing highly realistic images. This process is used in Generative Adverserial Networks.

Dropout layer. A neural network layer which is only partially connected, often with a stochastic dropout chance. This has been shown experimentally to improve neural network robustness in many architectures by reducing overfitting.

Epoch. A single complete training pass through all available training data, e.g. learning from all samples, or learning from all mini-batches, according to the training strategy. Multiple training epochs will typically be utilised although alternative strategies do exist.

Feed-forward network. A neural network composed of distinct 'layers', where the outputs of one layer never feed back into earlier layers. This avoids the needs for any iterative solver approaches and results in a very computationally efficient 'forward pass'.

Generative adversarial network. A two-part neural network architecture comprising a generator and a discriminator, which are co-trained to produce realistic outputs which are hard to distinguish from real-world data. The discriminator replaces the traditional loss function.

Generator model. A model which produces a synthetic example of a particular class, such as a synthetic image or synthetic language. Examples include language or image generation. These are used as part of Generative Adverserial Networks among other applications.

Global receptive field. Where every part of the input region can influence or stimulate a response in a model (e.g. a fully-connected neural network).

GPU. Graphical Processing Unit. A hardware device specialised for fast matrix operations, originally created to support computer graphics, particularly for games.

Gradient boosted decision tree. Also referred to as extreme gradient boosting. A random forest architecture which combines gradient boosting with decision tree ensembles.

Gradient boosting. An approach to model training where each additional ensemble member attempts to predict the cumulative errors of previously trained members.

Graph neural network. A class of neural networks designed to process data which is described by a graph (or tree/network) data structure.

Hidden layer. A layer which is intermediate between the input layer and the output layer of a network or tree structure. Hidden layers may be used to encode 'hidden variables' which are latent to a problem but not able to be directly observed, may constitute layers of a deep neural network, or may have other purposes.

https://doi.org/10.5194/egusphere-2023-350
Preprint. Discussion started: 17 April 2023

Hierarchical temporal aggregation. A mechanism of composing neural networks which are trained for different lead times to produce an optimal prediction at all time horizons.

Hierarchical temporal memory. Fundamentally different to hierarchical temporal aggregation. A complex deep learning architecture which uses time-adjacency pooling.

Hyperparameter. A parameter which is not derived via training. Examples include the learning rate and the model topology.

Hyperparameter search (or Hyperparameter optimisation). The process of determining optimal hyperparameters. This term may also be used to encompass the model selection problem. This process is automated in some cases.

Input layer. A layer which is composed of input nodes. Typically machine learning models will have one input layer at depth zero (i.e. with no preceding layers) and no input nodes at greater depths.

Input node. A node which represents an input or observed value.

K-fold cross-validation. A process of changing the validation and test data partitions during different iterations of training. This allows more of the training and validation data to be used while minimising overfitting. Some definitions include test data in this process but that is not ideal as the final test is no longer statistically independent.

Keras. A streamlined API for creating neural networks, integrated with Tensorflow. Originally built on the Theano framework for general mathematical evaluation. PyTensor and Aesara are related packages.

Kernel trick. For data sets which are not linearly separable, first multiplying the data by a nonlinear function in a higher dimension can result in a linearly separable higher-dimensional data set to which a simpler method can be used to model the data.

Knowledge based systems. A broad term from artificial intelligence meaning a system which that uses reasoning and a knowledge base to support decision making. Knowledge is represented explicitly and a reasoning or inference engine is used to arrive at new knowledge.

Layer. In tree or feed-forward network structures (e.g. decision trees and feed-forward neural networks), a layer refers to the set of nodes at the same depth within a network.

Leaf node. Aka output node. A node which does not have any child nodes.

Long short term memory network. A recurrent neural network architecture which processes sequences of tokens utilising a 'memory' component which can store information from tokens early in a sequence for use in prediction of tokens much later in a sequence. Typical applications include language prediction and time-series prediction of many kinds.

Loss function (also known as target function, training function, objective function, penalty score, error function, heuristic function, minimisation function). A differentiable function which is well-behaved, such that smaller values

https://doi.org/10.5194/egusphere-2023-350
Preprint. Discussion started: 17 April 2023

represent better model performance and larger values represent worse performance. An example would be the root-mean-squared-error of a prediction compared to the truth or target value.

Mini batch. A subset or 'mini batch' of the training data. Utilised for multiple reasons, including computational efficiency and to reduce overfitting. Aggregate error over a mini-batch is be learned rather than per-sample errors. This is the typical contemporary approach. See also training batch for in-depth discussion.

Neural network. A composition of 'input nodes', 'connections', 'nodes', 'layers', 'output layers' and 'activation functions' which are capable of complex modelling tasks. Originally designed to simulate human neural functioning and subsequently applied to a range of applications.

Node. Aka vertex. A small data structure in a network, tree or graph structure which is connected by edges. A node may represent a real-world value (such as a location) or an abstract value (such as in a neural network), or a decision threshold (such as in a decision tree).

One-hot vector. A vector of 1s and 0s, in which only one bit is set to 1. Typically produced during the first step in machine learning for language processing to create a word or feature embedding in a process called tokenisation or encoding. The length of the vector is commonly equal to the number of categories or symbols.

Output layer. A layer which comprises the leaf nodes or output nodes of a tree or network.

Perceptron. A single-layer neural network architecture for supervised learning of binary classification. Originally built as an electronic hardware device encoding weights with potentiometers and learning with motors. A multi-layer perceptron is the same thing as an ordinary neural network.

Persistence. Refers to the practice of treating some past observation or reanalysis (usually immediately prior to the starting point of the prediction period) as the future prediction and "persisting" this one state forward to every prediction lead time. The predictive model is then compared to this persistence prediction, essentially assessing the performance of the model against a steady state prediction. This, along with climatology, is often used as a baseline or bare minimum prediction to beat (i.e., a prediction better than persistence could be considered skilful vs persistence). This answers the question " is my model a better source of information than using what happened just before now?".

Physically-informed machine learning. Also known as physics-informed machine learning. Machine learning is considered physically informed when some aspect of physics is included in any way. Examples include adding a physical component to the loss function (e.g. to enforce conservation of physical properties) or using an activation function with physically realistic properties.

Predictive step, forward pass, evaluation. The process of calculating a model prediction from a set of input conditions. Distinct from the training phase or back-propagation step.

PyTorch. A widely adopted framework for neural networks in Python.

https://doi.org/10.5194/egusphere-2023-350
Preprint. Discussion started: 17 April 2023

Random forest. An architecture based on decision tree ensembles where each decision tree is initialised semi-randomly and an average of all models is used for prediction. This is typically more accurate than a single decision tree but less accurate than a gradient-boosted decision tree and so is now less-used. The term random forest is still commonly used when in fact the implementation is a gradient boosted decision tree.

Receptive field. The size or extent of a region in the input which can influence or stimulate a response in a model, e.g. the size of a convolutional kernel, the size of a sliding window

Recurrent network. A neural network which does pass the output from nodes of the network back into the input of others. Infinite recurrence is avoided by setting a specific number of iterations for the recurrence. These are often depicted in diagrams as separate layers but the implementation is through internal recurrent connections.

Regression. A model which attempts to diagnose or predict an exact value by statistically relating example input values to desired values.

Relevance vector machine. A sparse Bayesian model utilising the kernel trick in similar fashion to a support vector machine.

Representation error. Error which is introduced due to the inexactness of representing the real world in the model belief state. Examples may include topography smoothing, point-to-grid translations, model grid distortions near the poles, or the exclusion of physical characteristics which are not primary to the model.

Residual deep neural network. A very influential and innovative convolutional deep learning architecture which uses a similar concept to gradient boosting. Each layer of the deep network is taken to predict the residual error from the previous layers, with skip-connections from earlier layers allowing the training to occur without the issue of vanishing gradients.

Sample. A single training example (e.g. a row of data).

Scale invariance. A feature of a system, problem or model which means the results and behaviour are the same at any scale (e.g., the behaviour does not change if the inputs are multiplied by a common factor).

Scikit-learn. A popular Python library for machine learning which extends the SciPy framework.

(Stochastic) Gradient descent. An algorithm by which a neural network is trained using increasingly fine-scale adjustments to optimise the accuracy of network prediction. Utilised to find the local minimum of a differentiable function.

Supervised learning. Machine learning is considered 'supervised' when the data is labelled according to a category or target value. Classification data have an explicit labelled category. Regression data have an explicit value which is being predicted for.

Support vector machine. A classification model based on finding a hyperplane to separate data utilising the kernel trick.

https://doi.org/10.5194/egusphere-2023-350
Preprint. Discussion started: 17 April 2023

Tensor. Can be considered as a dense multi-dimensional array or matrix.

Tensorflow. A widely adopted framework for neural networks in Python.

Test/train/validate split. Available data is split into three portions. The training data is evaluated and used to update model weights. Validation data is evaluated during training and may be used for hyper-parameter search or to guide the researcher. Test data is independent (typically well-curated) data used for gold standard evaluation. In reality, validation data is sometimes used as test data, but this is not good practice. There are many considerations for test/train/validate splitting, such as statistical independence, representation of all classes, and bias. It is important to consider what the model is generalising "from" and "to", and ensuring appropriate examples are present in the training data and appropriate examples are reserved for validation and test.

Token. Tokenisation the process of mapping a symbolic or categorical sequence to a numerical representation which is suited to a sequence-based machine learning model. Commonly, a vector representation will be utilised for the token form. In language processing, either characters or words may be represented as tokens depending on the approach.

TPU. Tensor Processing Unit. A hardware device specialised for artificial intelligence and machine learning applications, in particular neural network operations.

Training batch (or simply batch). Multiple definitions apply and the use the term has evolved over time. Originally used in the context of learning from offline or saved historical data as opposed to online or realtime novel data. In this definition, the training batch is the saved data and refers to the whole training set. For example, a robot exploring a new environment in real-time must use an online learning technique and could not utilise batch training to map the unseen terrain. In more recent use, particularly in the areas of neural network learning, the offline saved data may be split into one or more batches (subsets). If one batch (the batch is the entire training set) is used, the aggregate errors for the entire training set are used to update the model weights and biases, and the learning algorithm is called batch gradient descent. If each example is presented individually, this is called online training (even when historical saved data is being used), the weights and biases are updated for from each individual example, and the algorithm used is stochastic gradient descent. If the data is divided into multiple batches, this is often referred to equivalently as mini batches. The weights and biases are aggregated over each mini batch. This is the most common contemporary approach, as it reduces overfitting and is a good balance of training accuracy, avoiding local minima, and computational efficiency.

Transfer learning. The process of training a model first on a related problem, and then conducting further training on a more specific problem. Examples could be training a model first in one geographical region and then in another; or training first at a low resolution then subsequently at a high resolution. This is frequently done to reduce training computation cost for similar problems by re-using the trained weights from a well-performing source model, or to overcome a problem of limited data availability by using multiple data sources.

https://doi.org/10.5194/egusphere-2023-350
Preprint. Discussion started: 17 April 2023

Transformer network. A token-sequence architecture which is capable of handling long-range dependencies. Initially applied to language processing, it has found effective application in image processing as an alternative to convolutional architectures.

Translation invariance. A feature of a system, problem or model which means the results and behaviour are the same after any spatial translation (i.e., the behaviour does not change if the inputs are shifted spatially to a new location).

U-Net. A type of convolutional neural network developed for biomedical image segmentation which has found broad application. In the contracting part of the network spatial information is reduced while feature information is increased. In the expanding part of the network, feature information is used to inform high-resolution segmentation. The name derives from the diagrammatic shape of the network forming a "U".

Unsupervised learning. Machine learning is considered 'unsupervised' when data is unlabelled. Examples include clustering, association and dimensionality reduction.

Vanishing Gradient. At the extremes, nonlinear functions used to calculate gradients can result in gradient values which are effectively zero. These small or zero values, once present in the weights and biases of a neural network, can entirely suppress information which would in fact be useful, and result in a local minima from which training cannot recover. This is particularly relevant to long token-series when long-distance connections are relevant. A variety of techniques including alternative activation functions, training weight decay, skip connections and attention mechanisms may each or all be utilised to ameliorate this issue.

Weights and biases. The parameter values for each neuron which represent the weighting factors to apply to the input values, plus an overall bias value for the node.

XGBoost. A popular Python library for gradient boosted decision trees.

Code Availability

No code was used in the preparation of this review.

Data Availability

No data was processed in the preparation of this review.

Author Contribution

COdBD researched and wrote Sections 3, 4, 5, 6 and 7, and provided review of sections 2, 8 and 9. TL researched and wrote sections 2, 8 and 9, and provided review of sections 3, 4, 5, 6, and 7. COdBD and TL co-wrote sections 1 and 10.

https://doi.org/10.5194/egusphere-2023-350
Preprint. Discussion started: 17 April 2023

Competing Interests

The authors declare that they have no conflict of interest.

Acknowledgements

The authors would like to thank Bethan White, Harrison Cook, Tom Dunstan and Karina Williams for their very helpful reviews of early versions of this manuscript.

References

Ackmann, J., Düben, P. D., Palmer, T. N., & Smolarkiewicz, P. K. (2020). Machine-learned preconditioners for linear solvers in geophysical fluid flows. *arXiv preprint arXiv:2010.02866*.

Adams, S. V., Ford, R. W., Hambley, M., Hobson, J. M., Kavčič, I., Maynard, C. M., ... & Wong, R. (2019). LFRic: Meeting the challenges of scalability and performance portability in Weather and Climate models. *Journal of Parallel and Distributed Computing*, *132*, 383-396.

Arcomano, T., Szunyogh, I., Wikner, A., Pathak, J., Hunt, B. R., & Ott, E. (2022). A Hybrid Approach to Atmospheric Modeling That Combines Machine Learning With a Physics-Based Numerical Model. *Journal of Advances in Modeling Earth Systems*, *14*(3), e2021MS002712.

Atkinson, S. (2020). Bayesian hidden physics models: Uncertainty quantification for discovery of nonlinear partial differential operators from data. *arXiv preprint arXiv:2006.04228*.

Bar-Sinai, Y., Hoyer, S., Hickey, J., & Brenner, M. P. (2019). Learning data-driven discretizations for partial differential equations. *Proceedings of the National Academy of Sciences*, *116*(31), 15344-15349.

Beucler, T., Rasp, S., Pritchard, M., & Gentine, P. (2019). Achieving conservation of energy in neural network emulators for climate modeling. *arXiv preprint arXiv:1906.06622*.

Beucler, T., Pritchard, M., Rasp, S., Ott, J., Baldi, P., & Gentine, P. (2021). Enforcing analytic constraints in neural networks emulating physical systems. *Physical Review Letters*, *126*(9), 098302.

Bhattacharya, K., Hosseini, B., Kovachki, N. B., & Stuart, A. M. (2020). Model reduction and neural networks for parametric PDEs. *arXiv preprint arXiv:2005.03180*.

Bi, K., Xie, L., Zhang, H., Chen, X., Gu, X., & Tian, Q. (2022). Pangu-Weather: A 3D High-Resolution Model for Fast and Accurate Global Weather Forecast. *arXiv preprint arXiv:2211.02556.*

Bihlo, A., & Popovych, R. O. (2022). Physics-informed neural networks for the shallow-water equations on the sphere. *Journal of Computational Physics*, *456*, 111024.

Bolton, T., & Zanna, L. (2019). Applications of deep learning to ocean data inference and subgrid parameterization. *Journal of Advances in Modeling Earth Systems*, *11*(1), 376-399.

https://doi.org/10.5194/egusphere-2023-350
Preprint. Discussion started: 17 April 2023

Breiman, L. (2001). Random forests. *Machine learning*, *45*(1), 5-32.

Brenowitz, N. D., & Bretherton, C. S. (2018). Prognostic validation of a neural network unified physics parameterization. *Geophysical Research Letters*, *45*(12), 6289-6298.

Brenowitz, N. D., & Bretherton, C. S. (2019). Spatially extended tests of a neural network parametrization trained by coarse-graining. *Journal of Advances in Modeling Earth Systems*, *11*(8), 2728-2744.

Brenowitz, N. D., Beucler, T., Pritchard, M., & Bretherton, C. S. (2020). Interpreting and stabilizing machine-learning parametrizations of convection. *Journal of the Atmospheric Sciences*, *77*(12), 4357-4375.

Brenowitz, N. D., Henn, B., McGibbon, J., Clark, S. K., Kwa, A., Perkins, W. A., ... & Bretherton, C. S. (2020). Machine learning climate model dynamics: Offline versus online performance. *arXiv preprint arXiv:2011.03081*.

Brenowitz, N. D., Perkins, W. A., Nugent, J. M., Watt-Meyer, O., Clark, S. K., Kwa, A., ... & Bretherton, C. S. (2022). Emulating Fast Processes in Climate Models. *arXiv preprint arXiv:2211.10774.*

Chantry, M., Christensen, H., Dueben, P., & Palmer, T. (2021). Opportunities and challenges for machine learning in weather and climate modelling: hard, medium and soft AI. *Philosophical Transactions of the Royal Society A*, *379*(2194), 20200083.

Chantry, M., Hatfield, S., Dueben, P., Polichtchouk, I., & Palmer, T. (2021). Machine learning emulation of gravity wave drag in numerical weather forecasting. *Journal of Advances in Modeling Earth Systems*, *13*(7), e2021MS002477.

Chase, R. J., Harrison, D. R., Burke, A., Lackmann, G. M., & McGovern, A. (2022a). A Machine Learning Tutorial for Operational Meteorology, Part I: Traditional Machine Learning. *arXiv preprint arXiv:2204.07492.*

Chase, R. J., Harrison, D. R., Lackmann, G., & McGovern, A. (2022b). A Machine Learning Tutorial for Operational Meteorology, Part II: Neural Networks and Deep Learning. *arXiv preprint arXiv:2211.00147.*

Chattopadhyay, A., Subel, A., & Hassanzadeh, P. (2020). Data-driven super-parameterization using deep learning: Experimentation with multiscale lorenz 96 systems and transfer learning. *Journal of Advances in Modeling Earth Systems*, *12*(11), e2020MS002084.

Chevallier, F., Chéruy, F., Scott, N. A., & Chédin, A. (1998). A neural network approach for a fast and accurate computation of a longwave radiative budget. *Journal of applied meteorology*, *37*(11), 1385-1397.

Clare, M. C., Jamil, O., & Morcrette, C. J. (2021). Combining distribution-based neural networks to predict weather forecast probabilities. *Quarterly Journal of the Royal Meteorological Society*, *147*(741), 4337-4357.

Cuomo, S., Di Cola, V. S., Giampaolo, F., Rozza, G., Raissi, M., & Piccialli, F. (2022). Scientific Machine Learning through Physics-Informed Neural Networks: Where we are and What's next. *arXiv preprint arXiv:2201.05624*.

De Bézenac, E., Pajot, A., & Gallinari, P. (2017). *Towards a hybrid approach to physical process modeling*. Technical report.

https://doi.org/10.5194/egusphere-2023-350
Preprint. Discussion started: 17 April 2023

Deng, J., Dong, W., Socher, R., Li, L. J., Li, K., & Fei-Fei, L. (2009, June). Imagenet: A large-scale hierarchical image database. In *2009 IEEE conference on computer vision and pattern recognition* (pp. 248-255). Ieee.

Dijkstra, H. A., Petersik, P., Hernández-García, E., & López, C. (2019). The application of machine learning techniques to improve El Niño prediction skill. *Frontiers in Physics*, 153.

Dosovitskiy, A., Beyer, L., Kolesnikov, A., Weissenborn, D., Zhai, X., Unterthiner, T., ... & Houlsby, N. (2020). An image is worth 16x16 words: Transformers for image recognition at scale. *arXiv preprint arXiv:2010.11929*.

Dueben, P. D., & Bauer, P. (2018). Challenges and design choices for global weather and climate models based on machine learning. *Geoscientific Model Development*, *11*(10), 3999-4009.

Dueben, P. D., Schultz, M. G., Chantry, M., Gagne, D. J., Hall, D. M., & McGovern, A. (2022). Challenges and Benchmark Datasets for Machine Learning in the Atmospheric Sciences: Definition, Status, and Outlook. *Artificial Intelligence for the Earth Systems*, *1*(3), e210002.

ECMWF. (2018). Ifs documentation (cy45r1). Retrieved from https://www.ecmwf.int/en/publications/ifs-documentation accessed 7th February 2023

Eyring, V., Bony, S., Meehl, G. A., Senior, C. A., Stevens, B., Stouffer, R. J., & Taylor, K. E. (2016). Overview of the Coupled Model Intercomparison Project Phase 6 (CMIP6) experimental design and organization. *Geoscientific Model Development*, 9(5), 1937-1958.

Flora, M., Potvin, C., McGovern, A., & Handler, S. (2022). Comparing Explanation Methods for Traditional Machine Learning Models Part 2: Quantifying Model Explainability Faithfulness and Improvements with Dimensionality Reduction. *arXiv preprint arXiv:2211.10378*.

Friedman, J. H. (2001). Greedy function approximation: a gradient boosting machine. *Annals of statistics*, 1189-1232.

Fuhg, J. N., Kalogeris, I., Fau, A., & Bouklas, N. (2022). Interval and fuzzy physics-informed neural networks for uncertain fields. *Probabilistic Engineering Mechanics*, *68*, 103240.

Gagne, D. J., McCandless, T., Kosovic, B., DeCastro, A., Loft, R., Haupt, S. E., & Yang, B. (2019, December). Machine learning parameterization of the surface layer: bridging the observation-modeling gap. In *AGU Fall Meeting Abstracts* (Vol. 2019, pp. IN44A-04).

Gagne, D. J., Christensen, H. M., Subramanian, A. C., & Monahan, A. H. (2020). Machine learning for stochastic parameterization: Generative adversarial networks in the Lorenz'96 model. *Journal of Advances in Modeling Earth Systems*, *12*(3), e2019MS001896.

Gagne, D. J., Chen, C. C., & Gettelman, A. (2020, January). Emulation of bin Microphysical Processes with machine learning. In *100th American Meteorological Society Annual Meeting*. AMS.

George, T., Gupta, A., & Sarin, V. (2008, December). A recommendation system for preconditioned iterative solvers. In *2008 Eighth IEEE International Conference on Data Mining* (pp. 803-808). IEEE.

https://doi.org/10.5194/egusphere-2023-350
Preprint. Discussion started: 17 April 2023

Gettelman, A., Gagne, D. J., Chen, C. C., Christensen, M. W., Lebo, Z. J., Morrison, H., & Gantos, G. (2021). Machine learning the warm rain process. *Journal of Advances in Modeling Earth Systems*, *13*(2), e2020MS002268.

Goodfellow, I., Pouget-Abadie, J., Mirza, M., Xu, B., Warde-Farley, D., Ozair, S., ... & Bengio, Y. (2020). Generative adversarial networks. *Communications of the ACM*, *63*(11), 139-144.

Goodfellow, I., Yoshua B., & Aaron C. (2016). Deep learning. *MIT press.*

Grigo, C., & Koutsourelakis, P. S. (2019). A physics-aware, probabilistic machine learning framework for coarse-graining high-dimensional systems in the Small Data regime. *Journal of Computational Physics*, *397*, 108842.

Ham, Y. G., Kim, J. H., & Luo, J. J. (2019). Deep learning for multi-year ENSO forecasts. *Nature*, *573*(7775), 568-572.

Ham, Y. G., Kim, J. H., Kim, E. S., & On, K. W. (2021). Unified deep learning model for El Niño/Southern Oscillation forecasts by incorporating seasonality in climate data. *Science Bulletin*, *66*(13), 1358-1366.

Han, Y., Zhang, G. J., Huang, X., & Wang, Y. (2020). A moist physics parameterization based on deep learning. *Journal of Advances in Modeling Earth Systems*, *12*(9), e2020MS002076.

Harder, P., Watson-Parris, D., Stier, P., Strassel, D., Gauger, N. R., & Keuper, J. (2022). Physics-informed learning of aerosol microphysics. *Environmental Data Science*, 1, e20.

Harris, L., Chen, X., Putman, W., Zhou, L., & Chen, J. H. (2021). A scientific description of the GFDL finite-volume cubed-sphere dynamical core.

Hastie, T., Tibshirani, R., Friedman, J. H., & Friedman, J. H. (2009). *The elements of statistical learning: data mining, inference, and prediction* (Vol. 2, pp. 1-758). New York: springer.

He, K., Zhang, X., Ren, S., & Sun, J. (2016). Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition* (pp. 770-778).

Hewamalage, H., Ackermann, K., & Bergmeir, C. (2022). Forecast Evaluation for Data Scientists: Common Pitfalls and Best Practices. *arXiv preprint arXiv:2203.10716*.

Hochreiter, S., & Schmidhuber, J. (1996). LSTM can solve hard long time lag problems. *Advances in neural information processing systems*, *9*.

Holloway, A., & Chen, T. Y. (2007, May). Neural networks for predicting the behavior of preconditioned iterative solvers. In *International Conference on Computational Science* (pp. 302-309). Springer, Berlin, Heidelberg.

Hornik, K., Stinchcombe, M., & White, H. (1989). Multilayer feedforward networks are universal approximators. *Neural networks*, *2*(5), 359-366.

Huang, Z., England, M., Davenport, J. H., & Paulson, L. C. (2016, September). Using machine learning to decide when to precondition cylindrical algebraic decomposition with Groebner bases. In *2016 18th International Symposium on Symbolic and Numeric Algorithms for Scientific Computing (SYNASC)* (pp. 45-52). IEEE.

https://doi.org/10.5194/egusphere-2023-350
Preprint. Discussion started: 17 April 2023

Karpatne, A., Atluri, G., Faghmous, J. H., Steinbach, M., Banerjee, A., Ganguly, A., ... & Kumar, V. (2017). Theory-guided data science: A new paradigm for scientific discovery from data. *IEEE Transactions on knowledge and data engineering*, *29*(10), 2318-2331.

Karunasinghe, D. S., & Liong, S. Y. (2006). Chaotic time series prediction with a global model: Artificial neural network. *Journal of Hydrology*, *323*(1-4), 92-105.

Kashinath, K., Mustafa, M., Albert, A., Wu, J. L., Jiang, C., Esmaeilzadeh, S., ... & Prabhat. (2021). Physics-informed machine learning: case studies for weather and climate modelling. *Philosophical Transactions of the Royal Society A*, *379*(2194), 20200093.

Keisler, R. (2022). Forecasting Global Weather with Graph Neural Networks. *arXiv preprint arXiv:2202.07575*.

Kim, J., Kwon, M., Kim, S. D., Kug, J. S., Ryu, J. G., & Kim, J. (2022). Spatiotemporal neural network with attention mechanism for El Niño forecasts. *Scientific Reports*, *12*(1), 1-15.

Kochkov, D., Smith, J. A., Alieva, A., Wang, Q., Brenner, M. P., & Hoyer, S. (2021). Machine learning–accelerated computational fluid dynamics. *Proceedings of the National Academy of Sciences*, *118*(21), e2101784118.

Krasnopolsky, V. M., Chalikov, D. V., & Tolman, H. L. (2002). A neural network technique to improve computational efficiency of numerical oceanic models. *Ocean Modelling*, *4*(3-4), 363-383.

Krasnopolsky, V. M., Fox-Rabinovitz, M. S., & Chalikov, D. V. (2005). New approach to calculation of atmospheric model physics: Accurate and fast neural network emulation of longwave radiation in a climate model. *Monthly Weather Review*, *133*(5), 1370-1383.

Krasnopolsky, V. M., Fox-Rabinovitz, M. S., & Belochitski, A. A. (2013). Using ensemble of neural networks to learn stochastic convection parameterizations for climate and numerical weather prediction models from data simulated by a cloud resolving model. *Advances in Artificial Neural Systems*, *2013*.

Kuefler, E., & Chen, T. Y. (2008, June). On using reinforcement learning to solve sparse linear systems. In *International Conference on Computational Science* (pp. 955-964). Springer, Berlin, Heidelberg.

Ladický, L. U., Jeong, S., Solenthaler, B., Pollefeys, M., & Gross, M. (2015). Data-driven fluid simulations using regression forests. *ACM Transactions on Graphics (TOG)*, *34*(6), 1-9.

Lam, R., Sanchez-Gonzalez, A., Willson, M., Wirnsberger, P., Fortunato, M., Pritzel, A., ... & Battaglia, P. (2022). GraphCast: Learning skillful medium-range global weather forecasting. *arXiv preprint arXiv:2212.12794*.

Lanthaler, S., Mishra, S., & Karniadakis, G. E. (2022). Error estimates for deeponets: A deep learning framework in infinite dimensions. *Transactions of Mathematics and Its Applications*, *6*(1), tnac001.

LeCun, Y., Bottou, L., Bengio, Y., & Haffner, P. (1998). Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, *86*(11), 2278-2324.

https://doi.org/10.5194/egusphere-2023-350
Preprint. Discussion started: 17 April 2023

Li, Z., Kovachki, N., Azizzadenesheli, K., Liu, B., Stuart, A., Bhattacharya, K., & Anandkumar, A. (2020a). Multipole graph neural operator for parametric partial differential equations. *Advances in Neural Information Processing Systems*, *33*, 6755-6766.

Li, Z., Kovachki, N., Azizzadenesheli, K., Liu, B., Bhattacharya, K., Stuart, A., & Anandkumar, A. (2020b). Neural operator: Graph kernel network for partial differential equations. *arXiv preprint arXiv:2003.03485*.

Li, Z., Kovachki, N., Azizzadenesheli, K., Liu, B., Bhattacharya, K., Stuart, A., & Anandkumar, A. (2020c). Fourier neural operator for parametric partial differential equations. *arXiv preprint arXiv:2010.08895*.

Lorenz, E. N. (1996). Predictability: A problem partly solved, in Proceedings of Seminar on Predictability, 4–8 September 1995.

Lu, L., Jin, P., & Karniadakis, G. E. (2019). Deeponet: Learning nonlinear operators for identifying differential equations based on the universal approximation theorem of operators. *arXiv preprint arXiv:1910.03193*.

McCulloch, W. S., & Pitts, W. (1943). A logical calculus of the ideas immanent in nervous activity. *The bulletin of mathematical biophysics*, *5*(4), 115-133.

Meyer, D., Hogan, R. J., Dueben, P. D., & Mason, S. L. (2022). Machine learning emulation of 3D cloud radiative effects. *Journal of Advances in Modeling Earth Systems*, *14*(3), e2021MS002550.

Mooers, G., Pritchard, M., Beucler, T., Ott, J., Yacalis, G., Baldi, P., & Gentine, P. (2021). Assessing the Potential of Deep Learning for Emulating Cloud Superparameterization in Climate Models With Real-Geography Boundary Conditions. *Journal of Advances in Modeling Earth Systems*, *13*(5), e2020MS002385.

Mudigonda, M., Kim, S., Mahesh, A., Kahou, S., Kashinath, K., Williams, D., ... & Prabhat, M. (2017). Segmenting and tracking extreme climate events using neural networks. In *Deep Learning for Physical Sciences (DLPS) Workshop, held with NIPS Conference*.

Nelsen, N. H., & Stuart, A. M. (2021). The random feature model for input-output maps between banach spaces. *SIAM Journal on Scientific Computing*, *43*(5), A3212-A3243.

Nguyen, T., Brandstetter, J., Kapoor, A., Gupta, J. K., & Grover, A. (2023). ClimaX: A foundation model for weather and climate. *arXiv preprint arXiv:2301.10343.*Nielsen, A. H., Iosifidis, A., & Karstoft, H. (2022). Forecasting large-scale circulation regimes using deformable convolutional neural networks and global spatiotemporal climate data. *Scientific Reports*, *12*(1), 1-12.

O'Gorman, P. A., & Dwyer, J. G. (2018). Using machine learning to parameterize moist convection: Potential for modeling of climate, climate change, and extreme events. *Journal of Advances in Modeling Earth Systems*, *10*(10), 2548-2563.

O'Leary, J., Paulson, J. A., & Mesbah, A. (2022). Stochastic physics-informed neural ordinary differential equations. *Journal of Computational Physics*, *468*, 111466.

https://doi.org/10.5194/egusphere-2023-350
Preprint. Discussion started: 17 April 2023

Ott, J., Pritchard, M., Best, N., Linstead, E., Curcic, M., & Baldi, P. (2020). A Fortran-Keras deep learning bridge for scientific computing. *Scientific Programming*, *2020*.

Palmer, T. (2020). A vision for numerical weather prediction in 2030. *arXiv preprint arXiv:2007.04830*.

Patel, R. G., Trask, N. A., Wood, M. A., & Cyr, E. C. (2021). A physics-informed operator regression framework for extracting data-driven continuum models. *Computer Methods in Applied Mechanics and Engineering*, *373*, 113500.

Pathak, J., Subramanian, S., Harrington, P., Raja, S., Chattopadhyay, A., Mardani, M., ... & Anandkumar, A. (2022). Fourcastnet: A global data-driven high-resolution weather model using adaptive fourier neural operators. *arXiv preprint arXiv:2202.11214*.

Peairs, L., & Chen, T. Y. (2011). Using reinforcement learning to vary the m in GMRES (m). *Procedia Computer Science*, *4*, 2257-2266.

Pincus, R., Mlawer, E. J., & Delamere, J. S. (2019). Balancing accuracy, efficiency, and flexibility in radiation calculations for dynamical models. *Journal of Advances in Modeling Earth Systems*, *11*(10), 3074-3089.

Psaros, A. F., Meng, X., Zou, Z., Guo, L., & Karniadakis, G. E. (2022). Uncertainty quantification in scientific machine learning: Methods, metrics, and comparisons. *arXiv preprint arXiv:2201.07766*.

Rasp, S., Pritchard, M. S., & Gentine, P. (2018). Deep learning to represent subgrid processes in climate models. *Proceedings of the National Academy of Sciences*, *115*(39), 9684-9689.

Rasp, S. (2020). Coupled online learning as a way to tackle instabilities and biases in neural network parameterizations: general algorithms and Lorenz 96 case study (v1. 0). *Geoscientific Model Development*, *13*(5), 2185-2196.

Rasp, S., Dueben, P. D., Scher, S., Weyn, J. A., Mouatadid, S., & Thuerey, N. (2020). WeatherBench: a benchmark data set for data-driven weather forecasting. *Journal of Advances in Modeling Earth Systems*, *12*(11), e2020MS002203.

Rasp, S., & Thuerey, N. (2021). Data-driven medium-range weather prediction with a resnet pretrained on climate simulations: A new model for weatherbench. *Journal of Advances in Modeling Earth Systems*, *13*(2), e2020MS002405.

Ravuri, S., Lenc, K., Willson, M., Kangin, D., Lam, R., Mirowski, P., ... & Mohamed, S. (2021). Skilful precipitation nowcasting using deep generative models of radar. *Nature*, *597*(7878), 672-677.

Rizzuti, G., Siahkoohi, A., & Herrmann, F. J. (2019, June). Learned iterative solvers for the Helmholtz equation. In *81st EAGE Conference and Exhibition 2019* (Vol. 2019, No. 1, pp. 1-5). European Association of Geoscientists & Engineers.

https://doi.org/10.5194/egusphere-2023-350
Preprint. Discussion started: 17 April 2023

Ronneberger, O., Fischer, P., & Brox, T. (2015, October). U-net: Convolutional networks for biomedical image segmentation. In *International Conference on Medical image computing and computer-assisted intervention* (pp. 234-241). Springer, Cham.

Russell S. & Norvig P (2021). *Artificial Intelligence: A Modern Approach* (Fourth Global Edition). Pearson Education.

Scher, S. (2018). Toward data-driven weather and climate forecasting: Approximating a simple general circulation model with deep learning. *Geophysical Research Letters*, *45*(22), 12-616.

Scher, S., & Messori, G. (2019). Weather and climate forecasting with neural networks: using general circulation models (GCMs) with different complexity as a study ground. *Geoscientific Model Development*, *12*(7), 2797-2809.

Sønderby, C. K., Espeholt, L., Heek, J., Dehghani, M., Oliver, A., Salimans, T., ... & Kalchbrenner, N. (2020). Metnet: A neural weather model for precipitation forecasting. *arXiv preprint arXiv:2003.12140*.

Taylor, J., & Feng, M. (2022). A Deep Learning Model for Forecasting Global Monthly Mean Sea Surface Temperature Anomalies. *arXiv preprint arXiv:2202.09967*.

Tibshirani, R., & Friedman, J. H. (2001). *The elements of statistical learning [electronic resource]: data mining, inference, and prediction: with 200 full-color illustrations*. Springer.

Tompson, J., Schlachter, K., Sprechmann, P., & Perlin, K. (2017, July). Accelerating eulerian fluid simulation with convolutional networks. In *International Conference on Machine Learning* (pp. 3424-3433). PMLR.

Turing, A. M., Computing Machinery and Intelligence, Mind, Volume LIX, Issue 236, October 1950, Pages 433–460, https://doi.org/10.1093/mind/LIX.236.433

Ukkonen, P., & Mäkelä, A. (2019). Evaluation of machine learning classifiers for predicting deep convection. *Journal of Advances in Modeling Earth Systems*, *11*(6), 1784-1802.

Ukkonen, P., Pincus, R., Hogan, R. J., Pagh Nielsen, K., & Kaas, E. (2020). Accelerating radiation computations for dynamical models with targeted machine learning and code optimization. *Journal of Advances in Modeling Earth Systems*, *12*(12), e2020MS002226.

Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., ... & Polosukhin, I. (2017). Attention is all you need. *Advances in neural information processing systems*, *30*.

Vlachas, P. R., Byeon, W., Wan, Z. Y., Sapsis, T. P., & Koumoutsakos, P. (2018). Data-driven forecasting of high-dimensional chaotic systems with long short-term memory networks. *Proceedings of the Royal Society A: Mathematical, Physical and Engineering Sciences*, *474*(2213), 20170844.

Walters, D., Boutle, I., Brooks, M., Melvin, T., Stratton, R., Vosper, S., ... & Xavier, P. (2017). The Met Office unified model global atmosphere 6.0/6.1 and JULES global land 6.0/6.1 configurations. *Geoscientific Model Development*, *10*(4), 1487-1520.

https://doi.org/10.5194/egusphere-2023-350
Preprint. Discussion started: 17 April 2023

Wang, S., Wang, H., & Perdikaris, P. (2021). Learning the solution operator of parametric partial differential equations with physics-informed DeepONets. *Science advances*, *7*(40), eabi8605.

Wang, X., Han, Y., Xue, W., Yang, G., & Zhang, G. J. (2022a). Stable climate simulations using a realistic general circulation model with neural network parameterizations for atmospheric moist physics and radiation processes. *Geoscientific Model Development*, *15*(9), 3923-3940.

Wang, S., Sankaran, S., & Perdikaris, P. (2022b). Respecting causality is all you need for training physics-informed neural networks. *arXiv preprint arXiv:2203.07404*.

Watson, P. A. (2022). Machine learning applications for weather and climate need greater focus on extremes. *Environmental Research Letters*, 17(11), 111004.

Watt-Meyer, O., Brenowitz, N. D., Clark, S. K., Henn, B., Kwa, A., McGibbon, J., ... & Bretherton, C. S. (2021). Correcting weather and climate models by machine learning nudged historical simulations. *Geophysical Research Letters*, *48*(15), e2021GL092555.

Werbos, P. (1974). Beyond regression: new tools for prediction and analysis in the behavioral sciences. *Ph. D. dissertation, Harvard University*.

Werbos, P. J. (1990). Backpropagation through time: what it does and how to do it. *Proceedings of the IEEE*, *78*(10), 1550-1560.

Weyn, J. A., Durran, D. R., & Caruana, R. (2019). Can machines learn to predict weather? Using deep learning to predict gridded 500-hPa geopotential height from historical weather data. *Journal of Advances in Modeling Earth Systems*, *11*(8), 2680-2693.

Weyn, J. A., Durran, D. R., & Caruana, R. (2020). Improving data-driven global weather prediction using deep convolutional neural networks on a cubed sphere. *Journal of Advances in Modeling Earth Systems*, *12*(9), e2020MS002109.

Weyn, J. A., Durran, D. R., Caruana, R., & Cresswell-Clay, N. (2021). Sub-seasonal forecasting with a large ensemble of deep-learning weather prediction models. *Journal of Advances in Modeling Earth Systems*, *13*(7)

Wikner, A., Pathak, J., Hunt, B., Girvan, M., Arcomano, T., Szunyogh, I., ... & Ott, E. (2020). Combining machine learning with knowledge-based modeling for scalable forecasting and subgrid-scale closure of large, complex, spatiotemporal systems. *Chaos: An Interdisciplinary Journal of Nonlinear Science*, *30*(5), 053111.

Wu, K., & Xiu, D. (2020). Data-driven deep learning of partial differential equations in modal space. *Journal of Computational Physics*, *408*, 109307.

Yamada, K., Katagiri, T., Takizawa, H., Minami, K., Yokokawa, M., Nagai, T., & Ogino, M. (2018, November). Preconditioner auto-tuning using deep learning for sparse iterative algorithms. In *2018 Sixth International Symposium on Computing and Networking Workshops (CANDARW)* (pp. 257-262). IEEE.

https://doi.org/10.5194/egusphere-2023-350
Preprint. Discussion started: 17 April 2023

Yang, C., Yang, X., & Xiao, X. (2016). Data-driven projection method in fluid simulation. *Computer Animation and Virtual Worlds*, *27*(3-4), 415-424.

Yeo, K., Grullon, D. E., Sun, F. K., Boning, D. S., & Kalagnanam, J. R. (2021). Variational inference formulation for a model-free simulation of a dynamical system with unknown parameters by a recurrent neural network. *SIAM Journal on Scientific Computing*, *43*(2), A1305-A1335.

Yuval, J., & O'Gorman, P. A. (2020). Stable machine-learning parameterization of subgrid processes for climate modeling at a range of resolutions. *Nature communications*, *11*(1), 1-10.

Yuval, J., O'Gorman, P. A., & Hill, C. N. (2021). Use of neural networks for stable, accurate and physically consistent parameterization of subgrid atmospheric processes with good performance at reduced precision. *Geophysical Research Letters*, *48*(6), e2020GL091363.

Zanna, L., & Bolton, T. (2020). Data-driven equation discovery of ocean mesoscale closures. *Geophysical Research Letters*, *47*(17), e2020GL088376.

Zhong, X., Ma, Z., Yao, Y., Xu, L., Wu, Y., & Wang, Z. (2023). WRF–ML v1. 0: a bridge between WRF v4. 3 and machine learning parameterizations and its application to atmospheric radiative transfer. *Geoscientific Model Development*, 16(1), 199-209.

Zhou, L., Lin, S. J., Chen, J. H., Harris, L. M., Chen, X., & Rees, S. L. (2019). Toward convective-scale prediction within the next generation global prediction system. *Bulletin of the American Meteorological Society*, 100(7), 1225-1243.