Does explicit time series modelling improve accuracy in short-term solar photovoltaic energy forecasting?

Ben Dixon

MSc Computer Science

Internal supervisors
Dr Maria Perez Ortiz and Professor John Shawe-Taylor

External supervisor
Jacob Bieker, Open Climate Fix

Submission date: 12 September 2022

[1]**Disclaimer:** This report is submitted as part requirement for the MSc Computer Science at UCL. It is substantially the result of my own work except where explicitly indicated in the text.

Abstract

Human-induced climate change has caused frequent and more intense weather events. Reaching levels of warming in the next few decades above 1.5C would increase multiple hazards and lead to significant humanitarian harms. But by reducing emissions, the rate of warming will increase slower and by less overall. Achieving this will require a transformation of the energy system and the rapid integration of low carbon energy sources into the electricity grid. Solar photovolatics (PV) are a low carbon energy source, but the need to keep the energy to the grid stable and the variability due to cloud cover means that intermittency is a challenge. Solar PV forecasting over time horizons between 30 minutes and 8 hours is often described a solar nowcasting, and is used to help anticipate peaks and troughs in demand. In this dissertation, I explore different machine learning approaches to time series forecasting of sequences using a dataset of UK satellite imagery and solar PV energy readings over a 1- to 4-hour time range. Convolutional neural networks, or CNNs, were developed for image processing, but can also be used for dealing with sequences as a block. Recurrent neural networks, or RNNs, such as long short-term memory (LSTM) models, deal explicitly with sequences by carrying forward state information across timesteps. I explore whether explicitly representing the task as a time series leads to greater predictive accuracy. I evaluate the accuracy of the results in light of the differences between the models. I also explore the relative prediction errors, the activation of the neural networks, and estimate the carbon emissions generated by and averted by the use of different model architectures. I consider the implications for machine learning and solar PV forecasting more generally.

Acknowledgements

Thanks to my internal and external supervisors, Dr Maria Perez Ortiz and Professor John Shawe-Taylor at UCL, and Jacob Bieker at Open Climate Fix, for all of their helpful feedback, suggestions, and encouragement. I am grateful to Jack Kelly and the rest of the organisation at Open Climate Fix and their commitment to open source data and code, which made this research possible. I am also grateful to the open source community, for providing free software for students and researchers everywhere. This dissertation was written using LaTeX in Overleaf, and my experiments were conducted on Google Colaboratory [19].

Note

This dissertation was done as part of my MSc Computer Science course at UCL. Previous to this research, my only experience of ML was in COMP0142 Machine Learning for Domain Specialists, and I had not looked at neural networks, CNNs, RNNs, or LSTMs before.

Glossary

Term	Meaning
CNN	Convolutional neural networks - a class of neural network designed to work with image data by searching for particular patterns, see 2.1
Conv3D	A convolution over a three-dimensional tensor, in this case where the third dimension is time
ConvLSTM	A variant of LSTM which performs a convolution operation inside the cell, explained in detail in 2.1
GPU	Graphical processing unit - a specialised electronic circuit for processing instructions which uses a parallel structure to perform many operations simultaneously, and so are often used in machine learning tasks
GHG	Greenhouse gas - the group of gases which increase global warming through the greenhouse effect, such as carbon dioxide and methane
LSTM	Long short-term memory - a class of neural network designed to work with sequence data and to carry information across multiple time steps, see 2.1
National Grid ESO	The energy service operator (ESO) for the UK, see 1.1
Nowcasting	Short-term forecasting, over a few hours in advance, used in this context to mean forecasting solar PV yield
NWP	Numerical weather prediction - a computationally intensive method for weather forecasting which uses multiple systems of differential equations based on physical principles
ML	Machine learning - an area of research in which computer models leverage data to improve performance on some set of tasks
OCF / Open Climate Fix	The external partner organisation for this project. One of OCF's main research areas is solar nowcasting. See 1.1
Persistence	A simple forecast made at time t, where the prediction for $t+1, t+2, ...$ is equal to the reading at $t-1$
PV	Photovoltaic - the conversion of light into energy
PV yield	The amount of energy produced through solar photovoltaics
RNN	Recurrent neural network - a class of neural network designed to work with sequence data which carries information from one time step to the next. See 2.1.
Solar altitude	The angle between the horizon and the line to the sun - zero degrees at sunset and sunrise, and at most 90 degrees at midday near the equator

Table 1: A glossary of terms used throughout the project

Contents

Chapter 1

Introduction

Chapter summary: I explain why short-term solar photovoltaic forecasting is an important problem (1.1), and highlight what question I aim to study (1.2). I define my goals for the project (1.3), and the project management approach I take (1.4). Finally I explain the structure of this report and how it answers the questions posed (1.5).

1.1 Problem statement: the importance and challenge of solar nowcasting

Climate change is a major global problem and humanity urgently needs to reduce emissions. As a result of anthropogenic greenhouse gas emissions, global warming is expected to reach 1.5C in the next few decades and will cause unavoidable increases in multiple climate hazards and present multiple risks to ecosystems and humans, but near-term actions that limit global warming to close to 1.5C would significantly reduce projected losses [22]. Global greenhouse gas emissions continue to rise, but the rate of global emissions growth each year has slowed.

Reducing emissions to safer levels of warming will require a transformation of our energy system. A range of future energy pathways have been modelled to project likely future emissions, which in turn have associated probabilities of levels of warming. All pathways that are associated with a greater than 50% chance of limiting warning to 1.5C, and all pathways associated with a greater than 67% chance of limiting warming to 2C necessitate rapid and deep and in most cases immediate GHG emission reductions in all sectors [23].

Providing energy from solar photovoltaics (PV) is a promising way to reduce emissions. Solar energy works through capturing energy from the sun's light, and transforming this into electricity which can be input to the grid. The lifecycle costs of solar PV energy generation are far less than energy generation from fossil fuels. A 2020 World Bank review concluded that there was more than sufficient solar PV energy potential to meet global electricity demand [3]. In the UK, there are currently approximately 12 Gigawatts of solar PV capacity in the UK, and this is expected to increase to between 28 and 81 Gigawatts by 2050 [8].

However, the uncertainty in solar PV yield makes it difficult to integrate solar into the electricity grid. The source of solar energy is inherently uncertain because clouds can obscure the sun, and clouds can become thicker, thinner, and their movement across the sky is hard to predict. On a sunny spring day in Great Britain, solar power can account for 30% of all the electricity produced

on the island [37]. If clouds suddenly form, in a few minutes, the power grid can lose much of the energy generated from solar panels.

The reason this uncertainty is a problem is that the grid must always be finely balanced. At each point in time, the demand and supply to the UK energy grid must be exactly matched at 50Hz [11]. If supply suddenly decreases, the energy grid operators must immediately provide backup power in order to avoid power outages. In order to guarantee stable energy supply, grid operators have to keep spinning-reserve online, because these generators take hours to spin up. These reserves are usually gas turbines. Keeping these turbines on standby causes gas to be burnt, releasing carbon dioxide into the atmosphere.

These factors create an incentive to forecast solar PV yield more accurately. If we knew what the energy coming into the grid would be, there would be less of a need to keep the reserves on standby, and every kilowatt could be used efficiently.

NationalGrid ESO is the electricity system operator for Great Britain. Since 2015, 10 gigawatts of solar have come online, and NationalGrid ESO operates from the Electricity National Control Centre, where they manage over 1bn data points a day and make around 200 despatch instructions to the 300 - 400 generators that participate in the balancing market every hour [9]. UK grid operators have already been surprised by unexpected weather events, such as when oversupply of PV yield on an unexpectedly sunny day sent energy prices negative for six hours in 2019 [43].

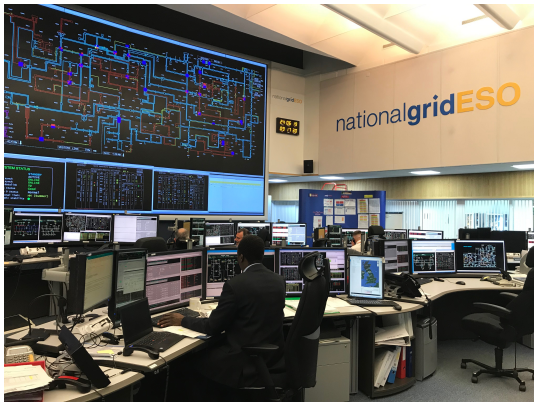


Figure 1.1: The UK's Energy National Control Centre, from [9]

National Grid ESO work with Open Climate Fix to improve the accuracy of their predictions [10] over short-term horizons. This forecasting work is described as 'nowcasting' to emphasize the short-term nature of the forecasts. NationalGrid ESO anticipates that improvements in solar nowcasting could be used to optimise grid management globally.

In 2021, Carolina Tortora, head of innovation and digital transformation at National Grid ESO said "Work like this has real impact – reducing forecasting errors and the need to keep costly fossil fuel plants ticking over... Open Climate Fix's nowcasting research has potential to further improve the forecasting capabilities of electricity system operators around the world." [37].

1.2 Research opportunities

There is a need to accurately forecast PV yield with a few hours notice. Existing methods for solar nowcasting are often computationally intensive and have low accuracy, and may not incorporate availiable information such as satellite images of the progression of clouds. From 2022, Open

Climate Fix have provided forecasting models to ESO National Grid incorporating the latest developments from machine learning [10].

Open Climate Fix experiments with different models to find which has the lowest mean absolute error, and then provides these models to National Grid ESO. Open Climate Fix experiment with a range of different approaches of using satellite images and other data for PV nowcasting. From 2019 to 2021, Open Climate Fix's basic production model was a convolutional neural network, or CNN. They are currently working on a power-perceiver using the perceiver architecture, which uses a transformer architecture with self-attention.

This dissertation aims to complement the PV nowcasting done by Open Climate Fix, by researching alternative methods for predicting PV yield, to examine and analyse the predictions and their errors. I cover this in more detail in the open questions section in the literature review in 2.3.

1.3 Goals and aims

1.3.1 Project goals

The objective of this report is to evaluate approaches in solar PV nowcasting. This requires understanding existing work through a literature review, posing hypotheses, defining experiments to answer those hypotheses, and analysing the results.

- **Task definition**: Define a specific PV nowcasting task. Open Climate Fix have experimented with different model designs on a wide range of tasks, over different time frames and with different geographical extent, and both predicting PV values and predicting the next image frame. I will review the main approaches used in computer vision and weather prediction and define a specific task which is feasible in the time available and provide useful information for solar PV nowcasting research.

- **Data collection**: Identify and validate appropriate data to assess this claim. There are multiple potential datasets, such as satellite images, numerical weather predictions, and different datasets of PV readings in the UK. I will select a dataset that allows me to perform the chosen task, is representative of the real-world application of UK energy forecasting, and is computationally feasible with the time and resources available.

- **Pre-processing and data exploration**: Transform the data into a form suitable for analysis with machine learning approaches. Machine learning works best with scaled and normalised features, and there may be missing or unrepresentative data which needs to be dealt with appropriately. I will also need to select a training, testing, and validation dataset.

- **Model selection**: Select models to predict the PV yield over an appropriate timescale. Open Climate Fix have used a Conv3D CNN, and I will review approaches in weather prediction, and choose models which I can use to test the hypotheses stated earlier.

- **Model evaluation and analysis**: Critically analyse and compare the results, and appraise this in relation to the work of Open Climate Fix and solar PV nowcasting generally. Use the models to derive predictions and compare these to true values, and see whether there are any systematic errors or patterns which indicate what the models are learning.

1.3.2 Personal aims

- Identify a valid and important computational problem, and narrow the scope such that I can address the problem in a dissertation
- Work with a combination of multiple real-world datasets
- Understand how neural networks and deep learning operate, going beyond what I have covered so far on my degree program
- Understand the main techniques and contemporary research in computer vision
- Deploy models with a high computational requirements, e.g. using GPUs and Google Colab

1.4 Project management

At the start of this dissertation in June 2021, I had only briefly studied applied machine learning as part of the course Machine Learning for Domain Specialists (COMP0142). Over the course of the following months, I needed to familarise myself with contemporary approaches in machine learning, computer vision, and find and complete a research project.

I considered following a rational planning / waterfall based model, defined as taking separate stages of specification, development, validation, and testing [42]. However, I wanted to get more immediate feedback and build up my machine learning skills starting with simple examples. For this reason I went for a more Agile approach, where the different processes described above are interleaved, with feedback from testing informing the task specification.

I did not develop an overall project plan, but instead focused on developing examples and implementing ideas from the literature as quickly as possible. I evaluated my progress based on the dates of my tests, and my overall project timeline is shown below.

Description	June				July				August				Sep	
Task definition														
Initial literature review														
Scoping call with OCF														
Researched OCF models														
Decided to compare CNNs with ConvLSTMs														
Defined scientific hypotheses														
Data collection and pre-processing														
Explored climatehack dataset														
Explored satellite dataset														
Explored pv dataset														
Cropped PV and aligned datasets														
Experiments														
Same-period PV prediction with CNN														
LSTM model														
CNN model														
ConvLSTM model for single value														
CNN model for single value														
Developed persistence model														
ConvLSTM and CNN models for sequence														
Multi-input models with past PV yield														
Hyperparameter tuning														
Report writing														
Literature review														
Analysis of results														
Interpretability of CNN activations														
Comparison with existing literature														
Report writing														
Feedback from OCF														

Table 1.1: A timeline showing my progress: the top-level goals matching those in my project goals are shown in text with dark green shaded cells, and the specific activities are shown underneath each goal, with indented text and lighter green shaded cells. Feedback from OCF is added at the bottom.

1.5 Structure of this report

This report is divided into different sections dealing with each of these stages, as described below.

- Chapter 2 - Literature review: Looks at different approaches to weather and PV nowcasting, and identify a research task that could be completed within the time available
- Chapter 3 - Datasets and preprocessing: details the sources and datasets, and the preprocessing to prepare them for analysis.
- Chapter 4 - Experimental design: Explains how I take models from the literature to test the proposed hypotheses.
- Chapter 5 - Results and discussion: summarises numerical results, plots predictions against observed values, analyses accuracy, and interprets activations in a convolution.
- Chapter 6 - Conclusion: reviews project overall, assesses how well it as able to address the questions raised earlier, implications for research and solar PV nowcasting, and proposed future research directions.

Chapter 2

Literature review

Chapter Summary: I briefly survey the literature on time series forecasting with a focus on machine learning and computer vision (2.1). I examine how these techniques have been applied in the domain of solar PV nowcasting and related fields (2.2). Drawing on these sections, I outline open research questions in (2.3) and develop two hypotheses to test these questions.

2.1 Time series forecasting approaches

Polynomial regression

Our task is to predict PV yield, a continuous value, and so we could deal with this as a regression task. Forecasting problems can sometimes be dealt with as a polynomial regression. For a simple linear regression of inputs $\mathbf{x}$ for which we minimise the weight vector $\mathbf{w}$, this would take the following form:

$$y(\mathbf{x}, \mathbf{w}) = w_0 + w_1 x_1 + ... + w_D x_D \tag{2.1}$$

If we maintain the constraint that the function is *linear in the weights* then we can extend the generality to a fixed nonlinear transformation of the inputs, and define this transformation as $\phi_j(\mathbf{x})$, which will output M transformed vectors of dimension *(1 X D)*, which we then multiply by $\sum_j^M w_j$, giving the following. We also simplify the representation of the bias term w_0 by defining a dummy basis function $\phi_0(\mathbf{x}) = 1$, which allows us to include $j = 0$ in the summation below.

$$\begin{aligned} y(\mathbf{x}, \mathbf{w}) &= \sum_{j=1}^{M-1} w_j \phi_j(\mathbf{x}) + w_0 \\ &= \sum_{j=0}^{M-1} w_j \phi_j(\mathbf{x}) \end{aligned} \tag{2.2}$$

The different possible values of $\mathbf{w}$ will give differing predictions, which we could compare to our training dataset using a loss function, such as mean squared error. I cover this process in detail in the following section. But it only makes sense to begin optimising if this functional form is likely to meaningfully capture the information in our inputs. Would a polynomial regression be a suitable

approach for our task of predicting PV yield from satellite images?

In short, probably not. A polynomial regression is likely to struggle with this task because the function determining PV yield is likely to be highly complex and to draw on many parts of the input vector. For example, consider two images with a similar PV yield forecast, one of a thin cloud covering the entire sky, and another one of a clear sky but with a dark cloudy patch moving toward our PV stations. These two images will have very different numerical form. There are many other examples which would look very different as vectors but might have similar PV yield, and a polynomial regression may not be able to find weights that minimise a transformation of the input pixels.

Artificial Neural Networks

Artificial Neural Networks (ANN) were developed to solve problems through multiple levels of abstraction, and were inspired by simplified models of biological neurons by neurophysiologist Warren McCulloch and the logician Walter Pitts [32] in 1943. An early application was the *Perceptron* in 1957 [38], and over time researchers stacked perceptrons on top of each other, with the outputs from one node forming the inputs into the next.

Each neuron works in a similar way to a polynomial regression, in that it is linear in the weights, but the difference with neural networks is that multiple neural networks can be stacked on top of each other. The output from one neuron is transformed by an activation function, σ, which standardises the outputs to a range so they can be passed as inputs to the next layer.

The models begin by starting with a series $j = 1, ..., M$ of combinations of the input variables. If the input variables are $x_1, ..., x_D$, we can represent a neuron in the first layer as follows.

$$a_j = \sigma(\sum_{i=1}^{D} w_{ji}^{(1)} x_i + w_{j0}^{(1)}) \tag{2.3}$$

The output of each of the j combinations is some activation, a_j. These values are then passed through an activation function, $h(.)$, such as the logistic sigmoid, σ_1, or the rectified linear function, σ_2, both shown below.

$$\begin{aligned} z_j &= \sigma_1(a_j) = \frac{1}{1 + \exp -a} \\ z_j &= \sigma_2(a_j) = \max(0, a) \end{aligned} \tag{2.4}$$

The output of the activation function is z_j, which is then input to the next layer of neurons, say $k = 1, ..., K$ neurons.

$$a_k = \sigma(\sum_{i=1}^{D} w_{ji}^{(2)} x_i + w_{j0}^{(2)}) \tag{2.5}$$

The output of the last year is $y(\mathbf{x}, \mathbf{w})$. By stacking these layers together, the model is able to find a general function, as shown in equation 2.2.

The crucial difference between ANNs and polynomial regressions is that ANNs exploit hierarchy, structure, and re-use of components. By stacking nodes on top of each other, nodes closer to the input data can learn low-level structures such as gradients of sequences or edges in images. The output of the lower nodes is then passed to the intermediate nodes. These nodes then combine intermediate-level features such as combinations of gradients in sequences, or basic shapes in im-

ages. In turn the outputs of these layers are fed into higher-level nodes, which learn features such patterns, faces, or generic images [6]. Neural networks are trained using backpropagation, devel-

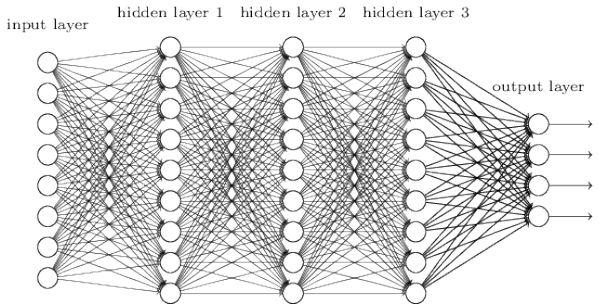


Figure 2.1: A neural network with three hidden layers, from Nielsen [33]

.

oped in 1985 [39]. Backpropagation consists of repeated forward and backward passes through the data. The forward pass consists of taking a inputs from the training data and passing it through the neural network, which starts with randomly initialised weights. At the output of the network, a comparison is done between the predicted outputs values and the actual output values. This comparison is captured in a *cost function*, and there are different choices on how to penalise divergences between the true value and expected values. The sum of the costs for all the observations passed through the neural network gives the *loss function*.

The next step is the backward pass, where the weights within each node are progressively adjusted such that the predicted values are closer to the true values. The fastest way to achieve this is by finding the gradients of each weight for the loss function for the neural network, then updating the weights in each neuron in the direction of the negative gradient, because the negative gradient is the direction of steepest descent on a surface. This starts on the last layer, and the weights are adjusted as described, and then the weights of the previous layer are adjusted in the same way, which leads to the neurons passing forward the information most helpful to the final layer. This process repeats until the input layer, at which point the process is complete.

Theoretically the loss function should be calculated as the sum of the cost for the entire training set, but this would require passing through the entire training set each time, and so stochastic gradient descent only passes in a batch of inputs which can form an approximation of the loss function to use the in process described above.

Neural networks can learn a variety of general structures, but in the examples below these structures will need to be located in the same place within the input. A standard multi-layer neural network may learn patterns based on specific locations within the input. However, for images, the exact location of a feature can vary, and for this reason a different type of network is often used.

Convolutional neural networks

A convolutional neural network, or CNN is a specific form of neural network used to detect patterns and often used in image tasks, first developed in 1998 [26]. CNNs work in such a way that they are able to recognise features regardless of their locations within the input data.

CNNs have hidden layers called convolutional layers which apply a number of different filters to an input. Each filter begins as a randomly initialised matrix, which is passed over the input,

where the size of the filter is specified as the kernel dimensions, such as (3x3). The filter will then pass over each area within the input which is the same size as the filter, and the dot product is computed between the value on the filter with the value of the input immediately below the filter.

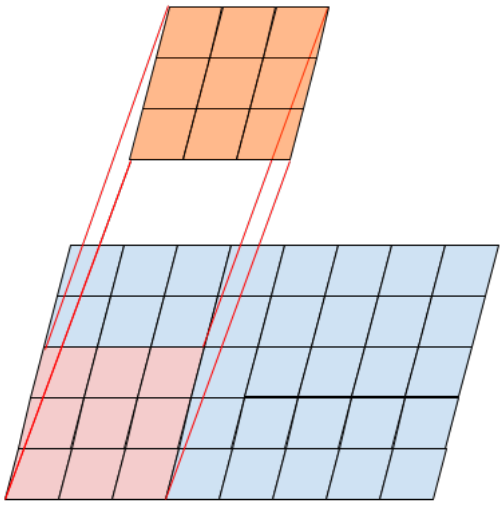


Figure 2.2: An example CNN with a (3x3) kernel part way through a convolution

A CNN begins with randomly initialised values in each of its kernels, and it will pass over the image and generate a cost and loss function as described above in backpropagation. For a CNN, the values that can change are the values within each filter. Over time through backpropagation this means the filters can take values that recognise particular features in the input, such as lines, edges, corner, or bright or dark areas. As these filters become tuned to the dataset, a high activation of a filter for a particular image implies that the shape the image is finding is present in the image, but it does not bound the template shape to appear in a particular location.

RNNs

Separately to CNNs, recurrent neural networks, or RNNs, have been developed to deal with sequence data. Feed-forward networks can only take a fixed-length input vector. Since sentences can be variable length, this motivated sequential processing in next word prediction.

RNNs have loops which allow for information to persist over time. This is done by passing state data between each prediction. For each prediction, an RNN takes input x_t and the past state h_{t-1}, and predicts output y_t and the next state h_t.

The internal state passed through each time step is based on a recurrence relation. The parameters in W are the same at each time step, and are learned during training.

$$\begin{aligned} h_t &= f_W(h_{t-1}, x_t) \\ h_{t-1} &= f_W(h_{t-2}, x_{t-1}) \end{aligned} \tag{2.6}$$

The activation function is *tanh* and this is applied to the product of two different matrices, which together with the weights of the prediction matrix, give three matrices to be learned.

The prediction at time t is given by a weight matrix applied to output of an activation function,

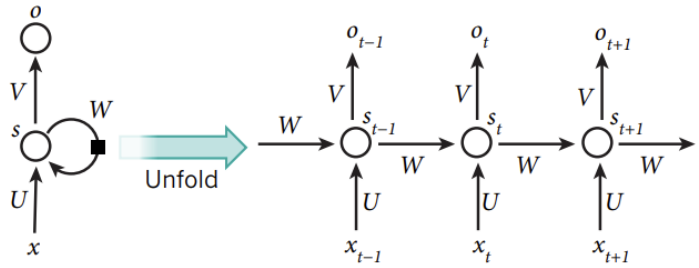


Figure 2.3: A representation a neuron in an RNN turning inputs to outputs, unfolded to show the same neuron at multiple points in time. The state is passed from left to right, and inputs are transformed to outputs moving from bottom to top. Taken from [27].

and that activation function includes both the input at t, which is x_t, and the *previous* hidden state h_{t-1}.

Feed-forward is trained by making a pass forward, then we back-propagate through the network and adjusting the parameters to minimise the loss. For RNNs, you need to work out the losses at each timestep in the sequence, and then finally across all timesteps back to the beginning of the sequence.

These means computing many gradients across time, and this can lead to exploding or vanishing gradients. This makes it harder to capture long-term dependencies. Using ReLU means we have a derivative either as 1 or 0, and so we don't have small gradients.

LSTMs

LSTMs were introduced by Hochreiter and Schmidhuber in 1997, to address the vanishing gradients problem, of remembering information from far back in the sequence [21]. The difference between LSTMS and RNNs is that LSTMs have separate information flows for short-term and long-term information. This allows weights to be learned to understand how long-term information affects the output, and a separate set of weights to form the associations related to short-term information.

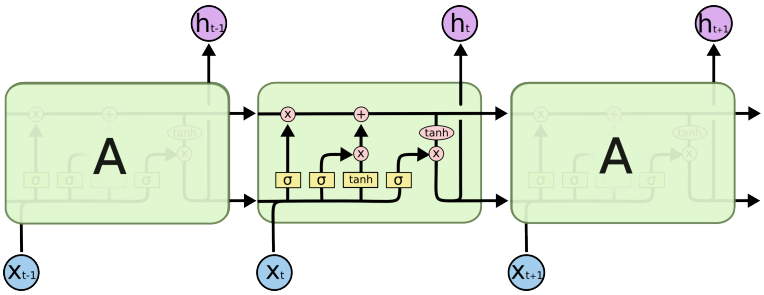


Figure 2.4: A visualisation of how an LSTM node is able to retain long-term information using gates, taken from [34]. Circles represent gates.

LSTMs use gates to control the flow of information, based on an neural net layer, and pointwise multiplication. Within each LSTM node, there are three gates which are combined in four stages : forget, store, update, and output. Olah [34] breaks down the four stages as follows. In the example below I use h_k to represent the output from the network at time $t = k$, following the notation used by Olah to match the diagram shown.

1. **The forget gate layer**: Take the current input x_t and previous output h_{t-1}, and learn a function which is input to the gate on the top left. We could interpret this as a function which tells us how much of the previous state we should keep.

$$f_t = \sigma(W_f \cdot [h_{t-1}, x_t] + b_f) \tag{2.7}$$

2. **Determine information to update the cell state**: A sigmoid layer determines which values we want to update, and the tanh layer creates a vector of candidate values, represented as $\widetilde{C}_t$.

$$\begin{aligned} i_t &= \sigma(W_i \cdot [h_{t-1}, x_t] + b_i) \\ \widetilde{C}_t &= tanh(W_C \cdot [h_{t-}, x_t] + b_C) \end{aligned} \tag{2.8}$$

3. **Update the cell state**: Use the outputs of the previous two functions, forgetting what we decided to forget, and updating what we decided to update, to update C_t. This uses the leftmost gate and the upper middle gate. Here, $*$ denotes a point-wise multiplication, because the output f_1 is a scalar between 0 and 1.

$$C_t = f_t * C_{t-1} + i_t * \widetilde{C}_t \tag{2.9}$$

4. **Generate a prediction**: We now make a prediction using the current state C_t, the previous prediction h_{t-1} and the current input x_t.

$$\begin{aligned} o_t &= \sigma(W_o \cdot [h_{t-1}, x_t] + b_o) \\ h_t &= o_t * tanh(C_t) \end{aligned} \tag{2.10}$$

LSTMs are designed to carry information across longer time frames than RNNs. By passing state information across time, separately to the prediction, and by having gates to determine how much of the state to update, and how much to forget for each prediction, they are able to use both short- and long- term information in their predictions.

Peephole connections

A modification of LSTMs was introduced by Gers and Schmidhuber [18] and was formalised by Graves [20]. Gers and Schmidhuber observed that each gate does not have a direct interaction with the cell state, which meant the relationships learned between each stage above was only indirectly linked. They added 'peephole' connections which meant each gate layer also took the previous step as input. A compact representation of the main equations is as follows. As before, $*$ denoting a point-wise multiplication, since the output of f_t is a scalar $[0, 1]$.

$$\begin{aligned} i_t &= \sigma(W_i \cdot [h_{t-1}, x_t, C_{t-1}] + b_i) \\ f_t &= \sigma(W_f \cdot [h_{t-1}, x_t, C_{t-1}] + b_f) \\ C_t &= f_t * C_{t-1} + i_t tanh(W_C \cdot [x_t, h_{t-1}] + b_c \\ o_t &= \sigma(W_o \cdot [x_t, h_{t-1}, C_{t-1}] + b_o) \\ h_t &= o_t * tanh(C_t) \end{aligned} \tag{2.11}$$

ConvLSTMs

The ConvLSTM was introduced in 2015 [41] to predict precipitation using radar images, combining ideas from convolutional neural networks and long short-term networks. ConvLSTMs are essentially peephole LSTMs, but which use the convolution operator rather than the Hadamard operator, or dot product.

I use $\otimes$ to denote the convolution operator. To emphasise the difference between vectors x_t, the authors use χ_t to represent images as 3D tensors where the last two dimensions are row and columns. Similarly the hidden outputs and states are also now 3D tensors and so are denoted with $\mathcal{H}_t$ and $\mathcal{C}_t$ to distinguish them.

$$\begin{aligned} i_t &= \sigma(W_{xi,hi} \otimes [\mathcal{H}_{t-1}, x_t] + W_{ic} \cdot \mathcal{C}_{t-1} + b_i) \\ f_t &= \sigma(W_{xf,hf} \otimes [\mathcal{H}_{t-1}, x_t] + W_{fc} \cdot \mathcal{C}_{t-1} + b_f) \\ C_t &= f_t * C_{t-1} + i_t tanh(W_C \otimes [x_t, \mathcal{H}_{t-1}] + b_c \\ o_t &= \sigma(W_{xo,ho} \cdot [x_t, \mathcal{H}_{t-1}] + W_{oc} \cdot \mathcal{C}_{t-1}] + b_o) \\ h_t &= o_t \otimes tanh(C_t) \end{aligned} \tag{2.12}$$

By including the convolutional operator as an input, this allows the model to take image data as input, to learn features through updating the weights on the convolutional filters', and then use this in the same way as LSTM models. As shown in the equations above, this means there are more sets of weights to learn, and the different sets of filters should focus on what information is useful from both a long- and short-term perspective.

2.2 Approaches to solar PV nowcasting and related fields

Weather prediction has been an important area of computational research since the 1960s. Numerical weather prediction (NWP) models solve a series of equations describing the physical laws that govern processes in the atmosphere, such as the Navier-Stokes equation. Model resolution has increased from 10-20km cells to around 1km [35], allowing more accurate predictions, but this comes with an increasing computational cost.

Since 2010, there have been significant improvements in machine learning as a result of massively parallel processing, convolutional neural networks, and large benchmark datasets. Deep neural networks have achieved breakthrough results in computer vision tasks [25], leading to increased interest from other domains including weather prediction.

Deep learning models may be simpler to design and faster to run than NWP models, as a result of parallel computations and not having to solve prognostic equations. The computational requirements of NWP models has led to increased interest in deep learning approaches, but deep learning models often come under criticism from the weather and climate research community for not having principled approaches, being hard to interpret, and potentially performing poorly under extreme conditions [40].

In a systematic review of 65 related publications between 2011 and 2020, Martins et al [31], found that most solar PV nowcasting work uses 'classical' approaches based on identifying cloud velocity and using this to forecast the impact of shade above a particular area. However, the authors found that deep learning approaches are becoming more popular over time, with some years seeing more deep learning approaches than classical methods.

Several recent deep learning approaches have yielded results competitve with NWP. In 2015, Shi

et al [41] develop a ConvLSTM model, combining of a CNN with a long short-term memory (LSTM) model, and demonstrate that it outperforms the state-of-the-art NWP optical flow algorithm in predicting precipitation over a 90 minute window in 6 minute increments.

Similarly, Bansal et al [4], who use data from 25 sites in the US over a year-long period. They find that using a combination of CNNs and LSTMs, they can develop point estimates that are highly accurate within 15 minutes compared with ground truth PV readings.

2.3 Open questions

There is comparatively less literature using deep learning techniques to predict solar PV yield than there is on precipitation or temperature, likely in part due to less accurate and timely PV yield data, possibly due to commercial and regulatory reasons, or the relative recency of the renewable sector [44].

In addition, there is debate over whether the explicit time series modelling in LSTMs and ConvLSTMs provides benefits above CNNs. Theoretically, since LSTMs have the ability to carry information across a sequence, their design suggests that they would perform better than CNN. However, in an influential paper[1] in 2018 Bai et al [2] argue that simpler CNNs can outperform LSTMs. In fact, they argue that 'We conclude that the common association between sequence modeling and recurrent networks should be reconsidered, and convolutional networks should be regarded as a natural starting point for sequence modeling tasks'.

To summarise, this leaves several gaps in the literature. How do different methods of solar PV nowcasting compare over timescales longer than 30 minutes? Are recurrent neural networks better than CNNs on sequence learning tasks, or should this association be reconsidered as Bai et al suggest?

Research focus and hypotheses

This dissertation aims to add to this discussion by explicitly comparing CNNs with LSTMs, and by extending the prediction period to four hours, and by examining the activations within the CNNs to identify which features are used to drive predictions.

There are two hypotheses I aim to test:

- **Hypothesis 1**: Explicitly modelling the data as a sequence improves performance.
- **Hypothesis 2**: Satellite images contain predictive information.

[1] By August 2022, the paper had over 2500 citations

Chapter 3

Datasets and preprocessing

Chapter summary: To address my research question, I needed to combine two geospatial time series datasets. I define two relevant datasets and how the measurements are captured (3.1). I then explain how I cropped and aligned these two datasets, dealt with missing data, and explored the data through visualiations (3.2). I summarise the overall pipeline (3.3), and explain how I generated batches for sequence modelling (3.3).

3.1 Data sources

In order to test my hypotheses, I needed to find open access data which I could use to test approaches for solar nowcasting using satellite images. After speaking with Open Climate Fix, I decided to combine two datasets, one of satellite images and the other of solar PV energy readings, which I could combine in order to test how the images could be used to forecast PV readings.

Satellite images

The satellite images are provided by The European Organisation for the Exploitation of Meteorological Satellites (EUMETSTAT) from the Meteosat-10 satellite which is in geostationary orbit 36,000km above the equator [12].

The Spinning Enhanced Visible and Infrared Imager (SEVIRI) [13] is the satellite's primary instrument and observes the Earth in 12 spectral channels. The satellite spins, taking an image of the northern third of the Meteosat disc every five minutes. The original EUMETSAT dataset contains data from 2008 to the present day in 12 spectral channels, and for a wide geographical extent.

Open Climate Fix have created the `eumetsat_uk_hrv` dataset [14] from a small subset of the entire SEVIRI dataset: it takes a single channel, the 'high resolution visible' (HRV) channel; it only contains data from January 2020 to November 2021; and it only includes data over the United Kingdom and over North Western Europe.

Even this reduced dataset is fairly large - the `eumetsat_uk_hrv` satellite image dataset is 570.22 GB, with frames taken every five minutes, with 1843 x 891 images. In order to make this more manageable, I reduced it down to a 128 x 128 crop focusing on Devon, and only chose images taken between 05:00 and 20:00 as these are typically daylight hours throughout the year, and as there would be minimal PV energy outside of this window for most of the year.

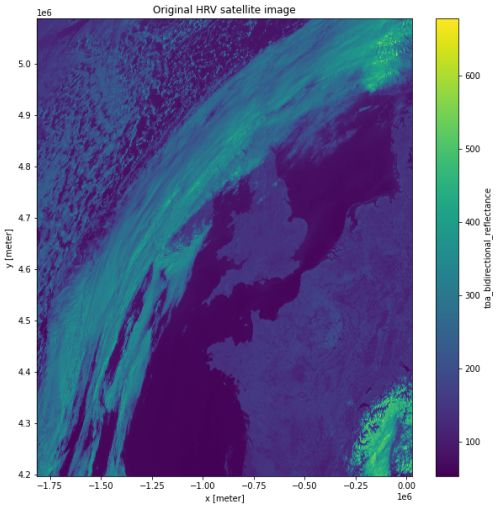


Figure 3.1: A satellite image of the UK, taken from the eumetsat dataset. Brighter sections indicate higher reflectivity.

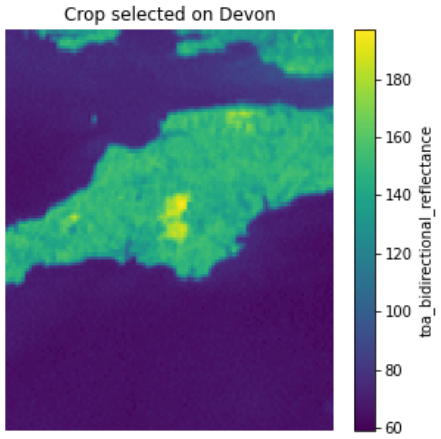


Figure 3.2: A crop of the eutmetsat dataset described above, focusing on Devon

PV solar generation data

Solar generation data refers to the yield from specific solar panels. Open Climate Fix have a dataset of from 1311 PV systems from 2018-01-01 to 2021-10-27, provided by a PV operator. The time series of solar generation is in 5 minutes chunks.

This data is collected from live PV systems in the UK. We have obfuscated the location of the PV systems for privacy.

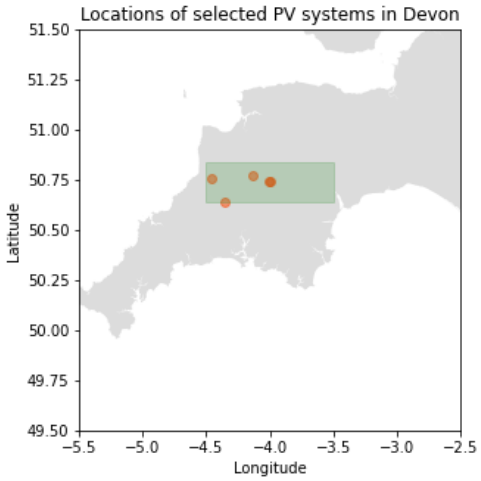


Figure 3.3: The locations of the selected solar photovolatic stations in Devon, highlighted as orange dots. The green box shows the search area.

3.2 Preprocessing and visualising

Aligning datasets

Once I had identified and cropped the datasets, I now had two datasets relevant to my task. However, the datasets were not aligned as the PV readings covered most of 2018 to 2021, and the satellite images were only from 2020 and 2021. I was also unsure whether there would be missing data. This might be caused when it is too dark for the satellite to take accurate readings, and there also might be times when the satellite is offline.

I verified that my datasets were spatially aligned by visualising them together.

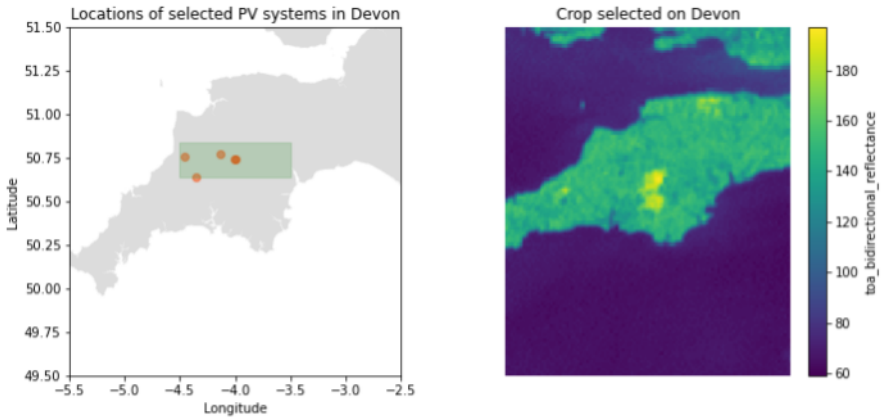


Figure 3.4: Side-by-side comparison of locations of satellite PV stations, and satellite image used in machine learning models.

I then noticed that the timesteps were at five minute increments both for satellite and PV readings. I removed PV readings from before 2020 from the PV readings, and then only chose satellite images for which there were PV readings. I looked at histograms of the average pixel

value across the image and saw there was a group of very low values.

I then set up a plotting function which would print to the screen a grid of the images at different intervals apart. This would allow me to quickly identify any discrepancies or missing data. An example of the preprocessed satellite images with a two hours spacing between images is in Fig 3.5.

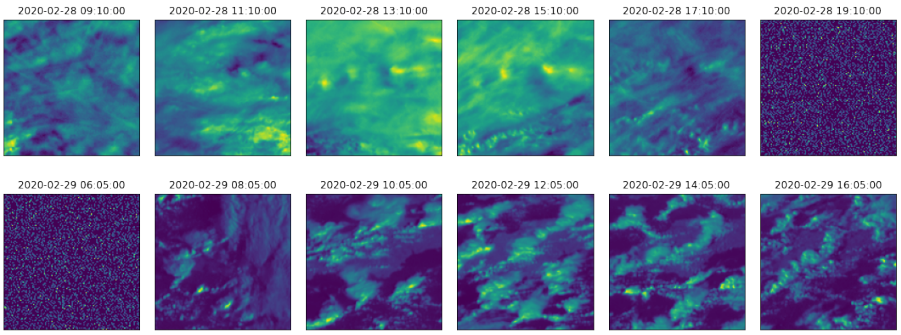


Figure 3.5: Preprocessed images, 2 hours apart

On manual inspection I saw that many of the images were effectively blanks, caused by the satellite trying to record an image while the particular section was dark. In the image above, this is visible on the top right image, taken at 19:10 on the 28th of February, and on the bottom left image, taken at 06:05 on 29th February. These images were unlikely to be informative and could be removed from the dataset. I had already cropped the satellite images to those taken between 05:00 and 20:00, but this still left many dark images taken during winter.

In order to preserve as much data as possible, I replicated an approach used by Open Climate Fix, and reduced the dataset to only those images taken when the solar altitude was above 10 degrees, using the Python package `suncalc` [5]. The solar altitude is a function of the date and time, and latitude and longitude. I set the latitude and longitude to the mean of the co-ordinates of the solar PV stations in my sample, giving values 50.73N, -4.19W.

When I examined this variable I thought it might be a useful potential feature. Over short time scales such as one day, solar altitude appears to be highly correlated with solar PV yield, as shown in Fig 3.6.

However, over longer time periods the correlation does not appear to be as strong, as PV yield is likely to be a complex function of solar altitude, and highly dependent on weather conditions such as cloud cover. This is visible in 3.7.

The satellite images have values in the range [0,255], so I normalised these to [0,1] by dividing by the 1 + the maxmimum value recorded across the dataset. I also normalised the PV readings by dividing by 1 + the maxmimum PV reading.

I also reduced the size of satellite image dataset to speed up training, from (128x128) to (64x64), by selecting a crop that still showed the relevant area, shown in Fig 3.8. I then did a final check on a grid of satellite images, along with the normalised PV readings, to check that they were correctly aligned.

3.3 Preprocessing pipeline

To summarise, I take data from two sources, crop both the relevant area, create test/train/validation datasets using non-contiguous days from each month, then align the PV and readings and

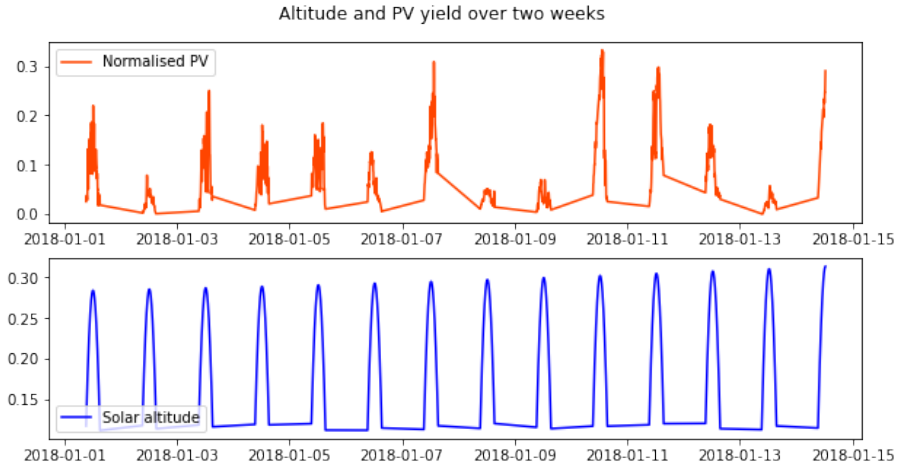


Figure 3.6: Solar altitude and PV yield over two weeks

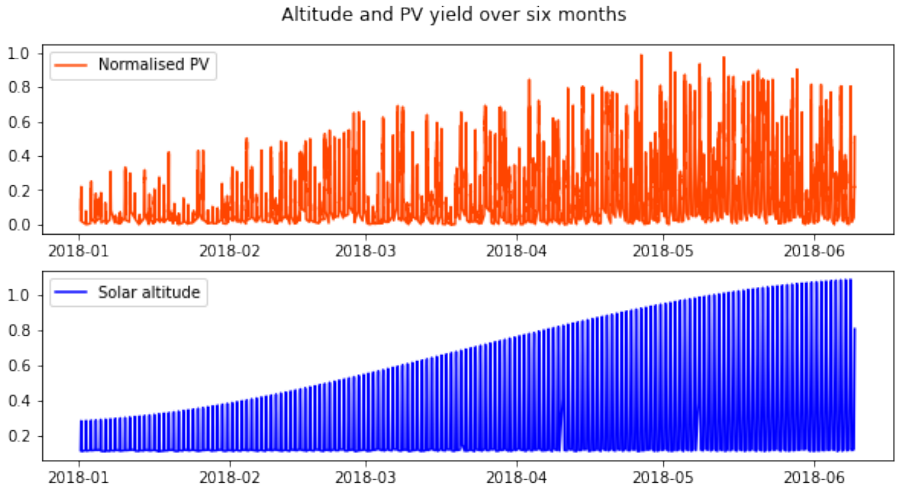


Figure 3.7: Solar altitude and PV yield over six months

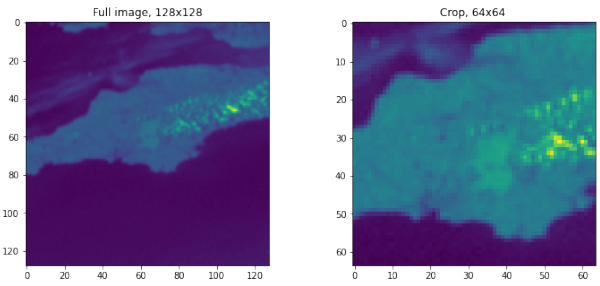


Figure 3.8: Smaller crop of satellite image, used to speed up training

satellite images and save them to Google Drive.

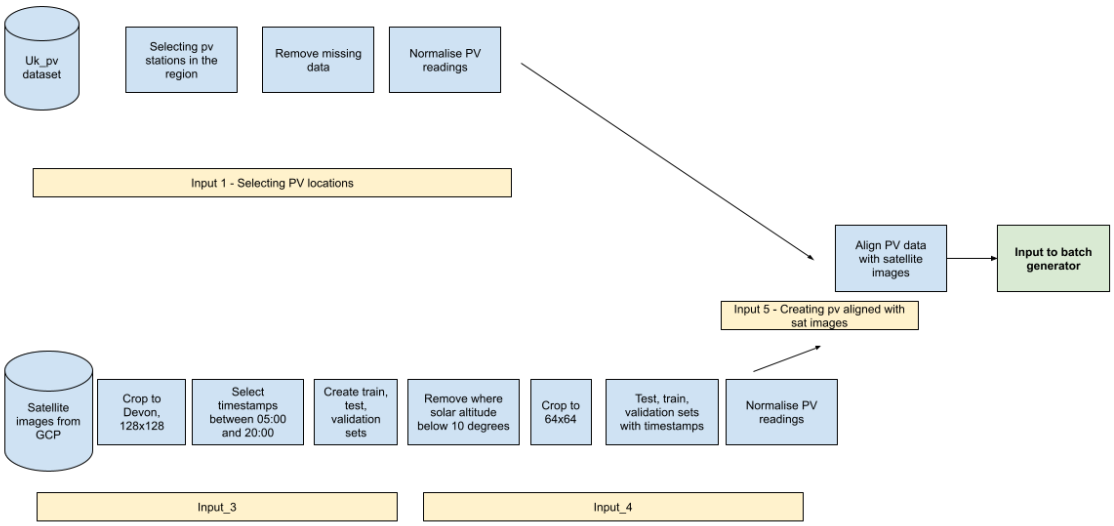


Figure 3.9: Preprocessing pipeline, taking satellite and PV readings, cleaning, aligning them, and inputting them to the batch generator.

Batch generation

Once the data is prepared, I form batches for each task. As the task can vary depending on the sequence length, this batching is done within each model file. The task is to take a sequence and to output a prediction sequence, and this required careful preparation of data, as results from one day should not be used to form predictions for the following day.

This is done using a custom dataloader I wrote, which takes the following steps:

1. Take the list of timestamps associated with a set of satellite readings, and return a dictionary of the days for each timestamp using `get_list_of_days`.
2. For each day, create a block which is the length of the number of readings for that day, using `get_blocks_of_timestamps_for_one_day`.
3. Concatenate all blocks together, using `get_full_block_list`.
4. Go through the block of timestamps, and make similar blocks to hold the satellite images, pv input reading, solar altitude inputs, and pv outputs, then populate the blocks with readings, using `get_sat_and_pv_blocks`.
5. Do this for each of the train, test, and validation datasets, then shuffle the blocks, expand the dimensions, then pad the input sequences so they match the output sequence length, using `dataloader`.

Chapter 4

Experimental design

Chapter summary: I explain which models I will use, drawing on my findings from the literature review. I start with a baseline model (4.1), then describe PV only models (4.1), and finally PV and satellite models (4.1). I compare a Conv3D based on a model used by OCF, with my own ConvLSTM design. Time series forecasting with autocorrelated data requires careful train/test/split generation, which I describe in 4.2. Next I cover hyperparameter turning and the loss function (4.3). Finally, I explain how these experiments can be used to evaluate the hypotheses stated earlier (4.4).

4.1 Selection of models

I also compare all models against a baseline model, where I use a persistence forecast.

Baseline models

A simple baseline model used in time series forecasting is the persistence model. This model simply predicts the next future value as equal to the most recent recorded value.

$$\hat{y}_t = y_{t-1} \tag{4.1}$$

For our task, we aim to predict a sequence of values. For example, if we take as input values from 09:00 to 9:55, our task is to predict PV yield from 10:00 to 10:55.

$$\hat{y}_{t=10:00} = \hat{y}_{t=10:05} = ... = \hat{y}_{t=10:55} = y_{t=09:55} \tag{4.2}$$

RNNs and CNNs - using prior PV only

We are predicting a sequence of values, and so our first two models take in PV data only. We consider two models, the first of which takes input as an LSTM, and the second as a CNN. Our models have multiple possible layers of CNN and LSTMs, followed by a fully connected layer with a minumum of 12 neurons to correspond to the 12 time sequence values predicted.

The first model is a CNN, which uses multiple levels to try to learn abstract features. After each convolutional layer, I use a MaxPooling layer to reduce the dimensionality and learn the essential features. This is followed by two dense layers to resize the output to predict a sequence of values. The CNN treats the sequence as one vector.

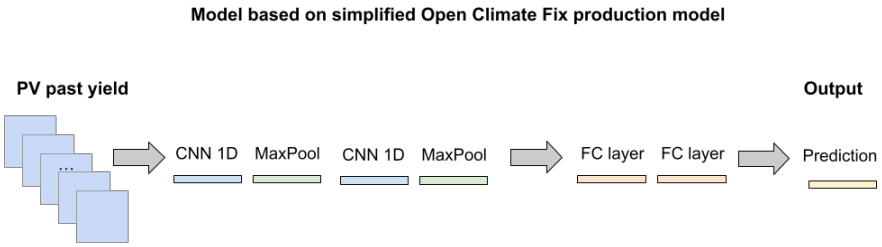


Figure 4.1: CNN model

The second model uses a recurrent neural network, an LSTM, again with multiple levels, to process the input as a sequence. The difference with the LSTM is that it passes each value through the sequence and updates the state as it progresses through the sequence. For the first two layers, it returns the full sequence of outputs, but on the last layer, it

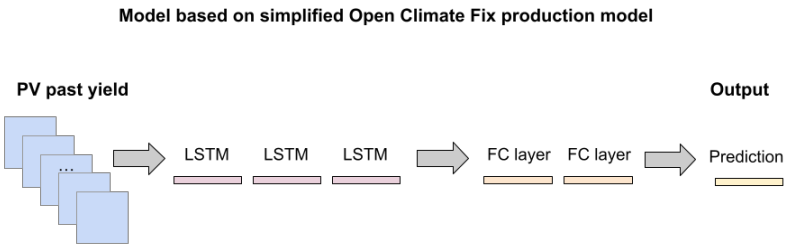


Figure 4.2: CNN model

RNNs and OCF Conv3D model - using both prior PV yield and satellite images

We now turn to models which include satellite data. For images, our main comparison is between CNN and ConvLSTM, and we are also interested in the comparison showing to what extent providing the prior PV data provides additional information. These models are based on the Conv3D architecture used by Open Climate Fix, comprised of four layers of Conv3D, and then fully connected layers combining the prior PV data. I use dropout [16] to reduce overfitting.

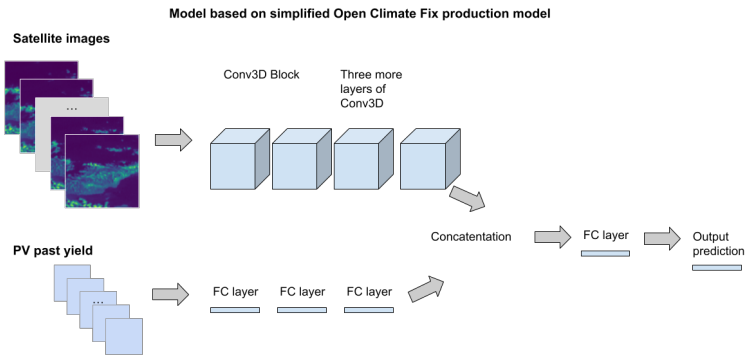


Figure 4.3: Basic OCF production model structure

4.2 Model training and dataset split

I trained the model on 22 months of data from 1/1/2020 to 7/11/2021, and split the data into train, test, and validation datasets. This is often done on a split of 80%:10%:10% between the three sets. When data is mutually independent, the subsets are often generated through random sampling.

However, as pointed out in Schultz et al [40], meteorological data constitutes a continuous time series with auto-correlation on different scales. As a result, randomly drawn samples could overlap, and information from the training dataset would leak into the test and validation datasets. This would impair our ability to discern whether the model was able to generalise, making hyperparameter tuning less helpful, and could lead to overly optimistic results as the test set effectively includes information used in training.

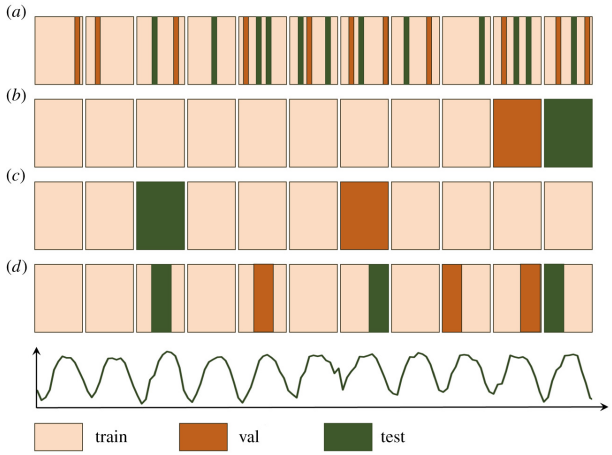


Figure 4.4: Different train-val-test strategies, from Schultz et al [40]

Figure 4.4 shows different splitting strategies for training, validation, and testing. Case (a) depicts random sampling, violating the independence assumption, and cases (b-d) show variants of random block sampling which avoid spurious correlations.

After speaking to Open Climate Fix, they advised an approach similar to case (b) above, but leaving a gap of a day between each section. The reason is that weather patterns often last for several days, and a contiguous split would in effect lead to data leaking from the test into the training data.

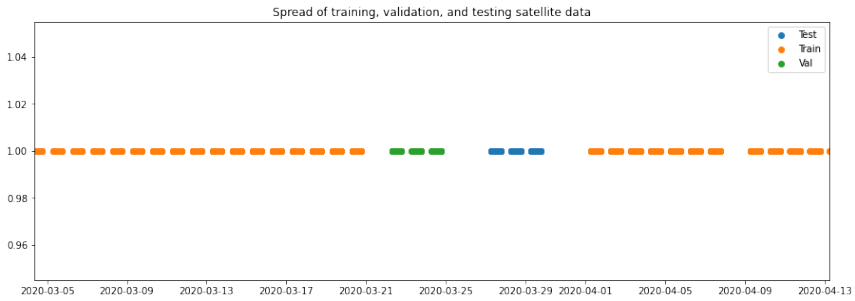


Figure 4.5: Train-val-test approach used

Figure 4.5 shows the split I used to generate my datasets. I balanced keeping the datasets

contiguous with data availability, and maintaining the conventional 80%:10%:10% split. I chose days 1 to 20 for training, days 22 to 24 for validation, and days 27 to 29 for testing.

4.3 Hyperparameter training and optimisation

There are several hyperparameters we can test in our models. For all models, we train the model until it converges on the training data, and vary the learning rate between [0.01, 0.001, 0.0001, 0.00001], and vary the dropout rate between [0.05, 0.5] in steps of 0.05.

For the CNNs and the ConvLSTMs, we vary the kernel size between [3,5,7]. For ConvLSTMs we vary the number of filters between [16,32,64]. For the fully connected layers we vary the number of nodes [12,24,48]. I trained the models until convergence using mean squared error, defined as follows.

$$MSE = \sum_{i=1}^{N} (\hat{\mathbf{y}}_i - \mathbf{y}_i)^2 \tag{4.3}$$

I use the Adam, or Adaptive Movement Estimation optimiser [24]. The Adam optimiser is an algorithm often used in deep learning and computer vision, as a modified form of gradient descent. The optimiser combines two techniques: gradient descent with momentum, and the nonuniform step sizes across different dimensions. Momentum means that the direction of gradient descent has some persistence between updates, to avoid gradient descent skipping over minima. Non-uniform step size allows for the weights to update by different amounts in different dimensions on the same step, but to still be influenced by the gradient of the loss function.

4.4 Tests for proposed hypotheses

In the literature review in chapter 2, I highlighted the debate over whether CNNs or RNNs are better at time series forecasting, and how deep learning multiple weather predictions incorporate satellite data. I aim to test these two ideas. I arrived at two hypotheses.

- **Hypothesis 1**: Explicitly modelling the data as a sequence improves performance.
- **Hypothesis 2**: Satellite images contain information which improves prediction accuracy.

I aim to test these hypotheses using the comparisons below. I will test the mean absolute error and root mean square error for each model, over a range of time horizons similar to those used in solar PV nowcasting in practice - 1, 2, 3, and 4-hour time series predictions. I will predict a sequence of values because grid operators are interested in the changing shape of the energy supply, not just a single point value.

Below are the comparisons used in each hypothesis. This will provide me with two sets of data points for each comparison, and I will be able to perform an ablation study by comparing the effectiveness with and without satellite imagery.

- **Hypothesis 1 (sequences)**: Compare 2D CNNs and LSTMs, then compare 3D CNN and ConvLSTM
- **Hypothesis 2 (satellite images)**: Compare 2D CNN and 3D CNN, then compare LSTM and ConvLSTMs

Chapter 5

Results and analysis

Chapter summary: I start comparing the performance of the models across the tasks (5.1). I evaluate the hypotheses in light of this evidence (5.2). Are there systematic errors in the predictions? Does it perform worse at some times than others? I explore patterns in the prediction errors (5.3). And how does the model derive its predictions? I visualise activations in a similar convolutional network in CNN (5.4). Finally, was it all worth it? I tally potential emissions generated and averted (5.5).

5.1 Results

In this section I present the results of the experiments. The task was to predict a sequence of values of variable lengths, from 1 hour to 4 hours. The charts below show MSE and RMSE by model over different prediction windows.

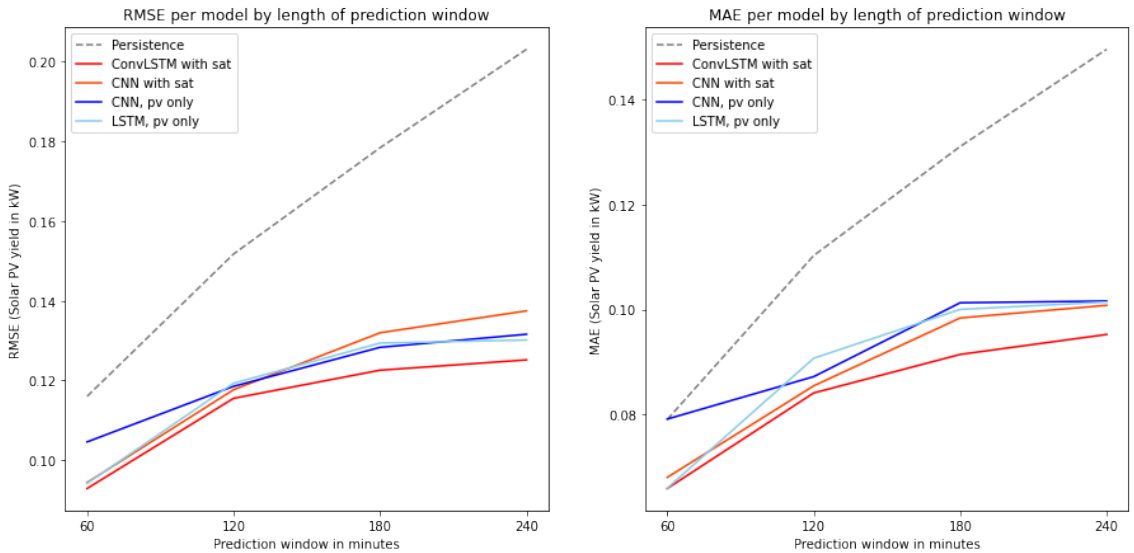


Figure 5.1: Charts comparing the RMSE and MAE for selected models, over a time period between 60 and 240 minutes. ConvLSTMs outperform other models, especially over longer time scales but otherwise the results are fairly mixed.

Key findings

The first finding is that all machine learning models presented outperform the baseline of a persistence model. The performance of this model also worsens over time as solar PV yield tends to vary more over several hours than shorter time periods.

Over time horizons up to 120 minutes, or 24 prediction frames, all models except the PV only CNN (in dark blue) have a similar performance on both metrics. Surprisingly, even the PV only LSTM has performance competitive with the models which also take satellite images as input.

As the time range increases past two hours, the ConvLSTM outperforms the CNN and all other models. On the RMSE measure, the CNN with satellite input actually performs worse than the two models which take satellite images as input.

		Baseline	PV only		Sat and PV	
Time	Metric	Persistence	CNN	LSTM	Conv3D	ConvLSTM
60 mins	RMSE	11.60%	10.46%	9.42%	9.44%	**9.29%**
	MAE	7.91%	7.92%	**6.60%**	6.81%	**6.60%**
120 mins	RMSE	15.18%	11.86%	11.93%	11.77%	**11.55%**
	MAE	11.03%	8.72%	9.07%	8.55%	**8.41%**
180 mins	RMSE	17.84%	12.83%	12.94%	13.20%	**12.26%**
	MAE	13.10%	10.13%	10.00%	9.84%	**9.15%**
240 mins	RMSE	20.32%	13.16%	13.02%	13.75%	**12.52%**
	MAE	14.95%	10.16%	10.14%	10.08%	**9.53%**

Table 5.1: Prediction error over different time horizons, with best results highlighted in bold. ConvLSTM is the best performing model.

5.2 Evaluation of hypotheses

5.2.1 Hypothesis 1 : Time series modelling improves accuracy

The first hypothesis is that modelling the data explicitly as a time series would improve prediction accuracy. The difference between recurrent neural networks and convolutional neural networks is that recurrent networks take both their inputs at the current time step, and one or more state vectors from the previous time step. For a LSTM network, there are two vectors which $\mathbf{h}_t$ and $\mathbf{c}_t$, which can be interpreted as short-term state and long-term state respectively [17].

For the PV only inputs, taking the four time periods and two error metrics, the LSTMs outperform CNNs on five out of eight comparisons. In three of these examples, the difference is less than 0.2%. For the PV and satellite inputs, the ConvLSTM model, which uses a recurrent architecture, outperforms the 3D CNN model on eight out of eight metrics. On this comparison the performance difference increases over time, from 0.2% differences at 60 and 120 minutes to 1% differences at 180 and 240 minutes.

This provides moderate evidence that the recurrent network approach outperforms convolutions, and that the state and memory used in explicit time series modelling helps to improve predictive accuracy. The finding seems to be more true over longer time periods and when the models include satellite images as inputs.

5.2.2 Hypothesis 2 : Satellite images improve predictions

The second hypothesis is that satellite images contain predictive information, so that by including satellite data as an input, we can improve our predictions. The theoretical justification would be that the main uncertainty in short-term PV yield is from cloud movement, and so if we can anticipate their movement using a sequence of satellite images, the model is able to exploit this information to improve predictions.

Comparing CNNs between those with PV only input, and those with PV and satellite input, for the four time horizons and two error metrics, on six out of eight comparisons, the models which take satellite input have a better performance. The differences are variable between 1% and 0.1% with no clear trend.

Comparing the recurrent architectures - the LSTM against the ConvLSTM - on seven out of eight comparisons the satellite image models perform better, plus one comparison where the results are tied. The improvement is fairly stable around 0.5%.

Overall this is also moderate evidence that satellite images improve predictions, but this seems more robustly true when the comparison leverages the recurrent network architecture of the LSTM.

5.3 Analysing the predictions

In this section I explore the predictions for the best model, the ConvLSTM model, over a one-hour time horizon. I begin by comparing the predictions against true values for three randomly chosen days in the test set.

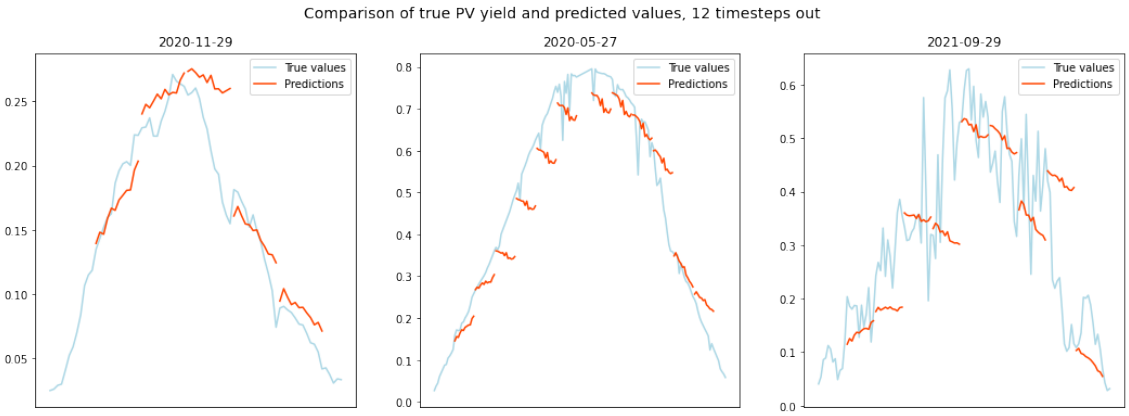


Figure 5.2: Predicted and observed PV yield across three randomly chosen days in the test set. Predictions an hour ahead are in orange and observed values in light blue. Note the variation in y values (solar yield) across days. The model tends to under-predict sunny and cloudy days.

Interestingly, it appears that the ConvLSTM generally predicts the overall gradient of the prediction correctly - if the prediction window is getting brighter or darker. However, in the middle day shown above, the model consistently predicts that the PV yield will decrease. The middle day has PV readings up to 75% of the two-year maximum, and is much brighter than the other two examples which peak at around 30% and 60%. This suggests that the model does not deal well with sunnier days, and might under-predict on those days.

The plot below shows the total relative prediction errors per mean prediction error over the sequence, across a two-year period. Since there are 12 predicted values with a range of [0,1], the largest possible magnitude of error is 12. The models optimise for mean square error, which is not sensitive to the sign of the error. This plot shows the direction of the error.

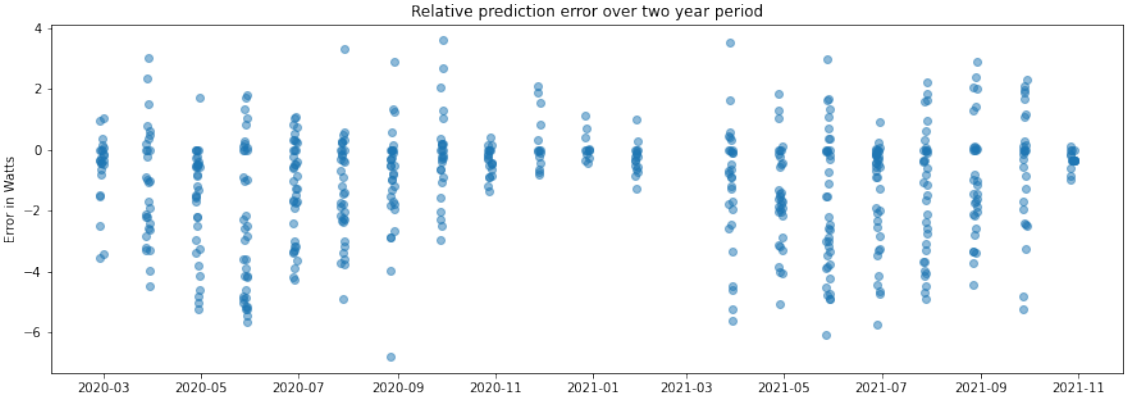


Figure 5.3: Plot of relative prediction error over two years - mean error over sequence. Overall the model tends to have a negative error - unpredicts PV yield. This is more extreme in summer, with smaller errors in winter.

It appears that the model tends to have the errors with the largest magnitude in summer, lower magnitude errors in November, December, and January. Since the test set days are three days chosen from the end of each month, the values appear in lines which span the test set ranges.

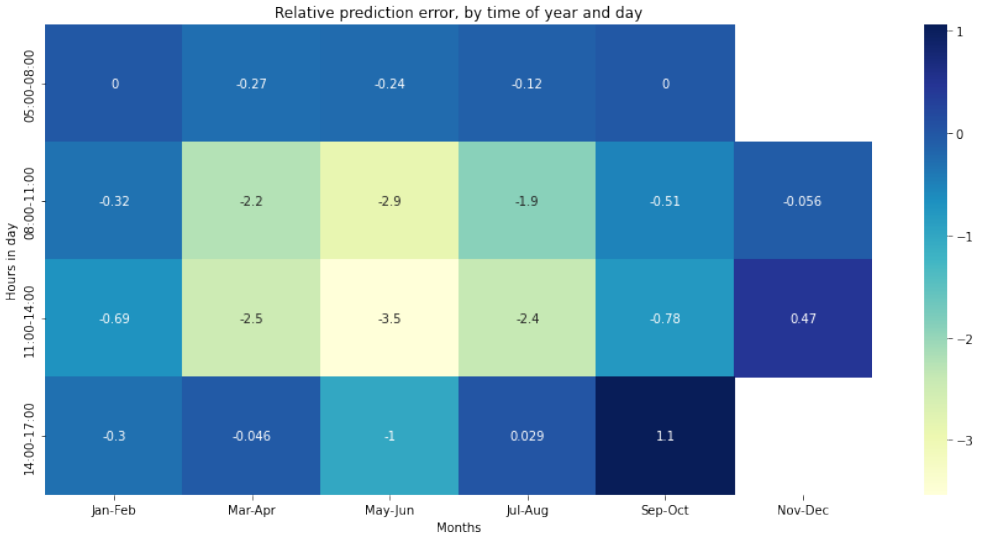


Figure 5.4: Heatmap of relative prediction error - mean error over sequence

Finally this heatmap represents the same data shown above. We can see that the model tends to under-predict mostly strongly during the early afternoon on summer days, which further shows that the model tends to find high PV yield and bright conditions anomalous, and reverts towards mean PV values.

5.4 CNN Interpretability

A frequent criticism of neural networks is that it is hard to interpret how they determine their results. In this section, I try to examine the activations in a similar neural network, to understand what drives the results, using an filter visualisation approach inspired by Chollet [6].

I have investigated the relationship between a sequence of observations and a sequence of pv readings. Interpreting the results would require viewing the activations across the sequence. For simplicity, I have trained a similar but simpler neural network to infer a single PV reading from a single image. The model architecture is similar to the CNN, and is comprised of four convolutional layers, but with only 16 filters per convolution, again for simplicity. After experimentation using ReLU, I discovered most of the images were dark. In order to make the activations easier to see, I used Leaky ReLU as the activation function, so a wider range of activation values would be visible. Leaky ReLU [29] shows small values for the gradient when the activation is less than zero.

I trained this model for 10 epochs on a randomly selected sample of 1000 satellite images, and it achieved a test set performance of a 2% mean square error. I select three images from the test set, which have different characteristics and will hopefully lead to different activations. I will refer to these as three different cases, which our CNN predicts fairly accurately.

- **Case A**: The left hand image is from a morning in summer and shows a clear sky and the land area is in clear relief to the sea, which has a high standardised PV reading of 0.60.
- **Case B**:The middle image has comparatively less contrast between land and sea, and is taken from early in the morning in spring, and is likely to be darker overall. Standardised PV reading around 0.09.
- **Case C**: The right image is taken from a summer morning, at a similar time to Case A but a different year. It has patchy cloud cover across the image, leading to a PV reading of 0.25.

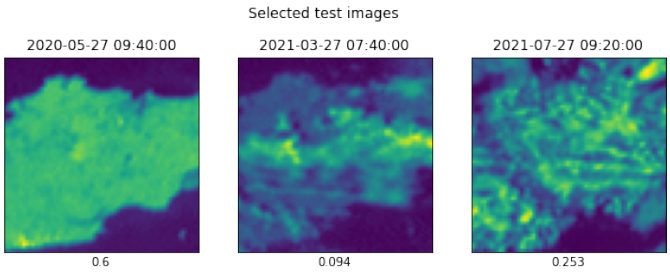


Figure 5.5: Select test images for interpretability

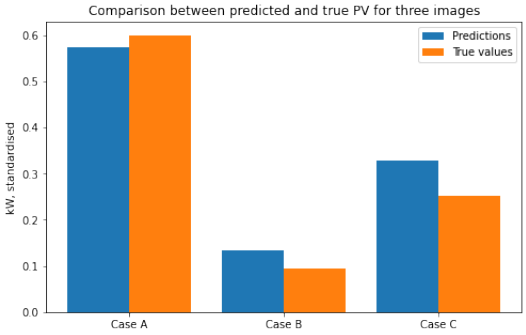
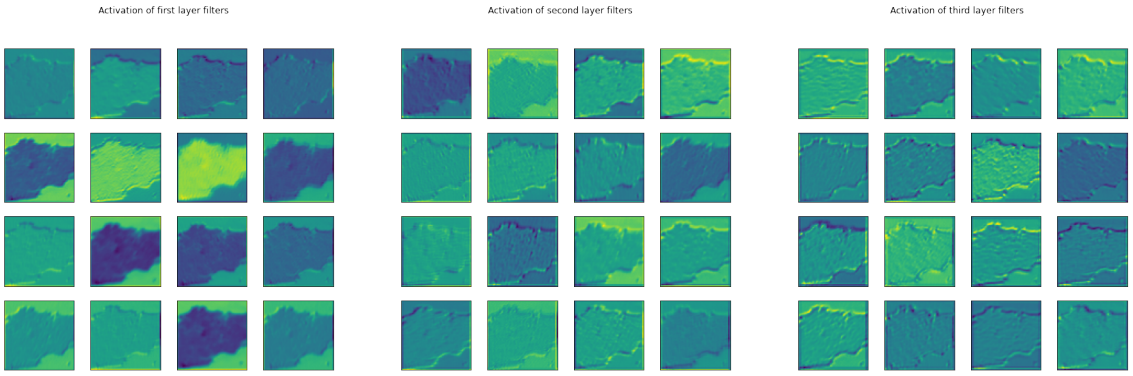


Figure 5.6: Comparison between predicted and actual PV yield for three examples

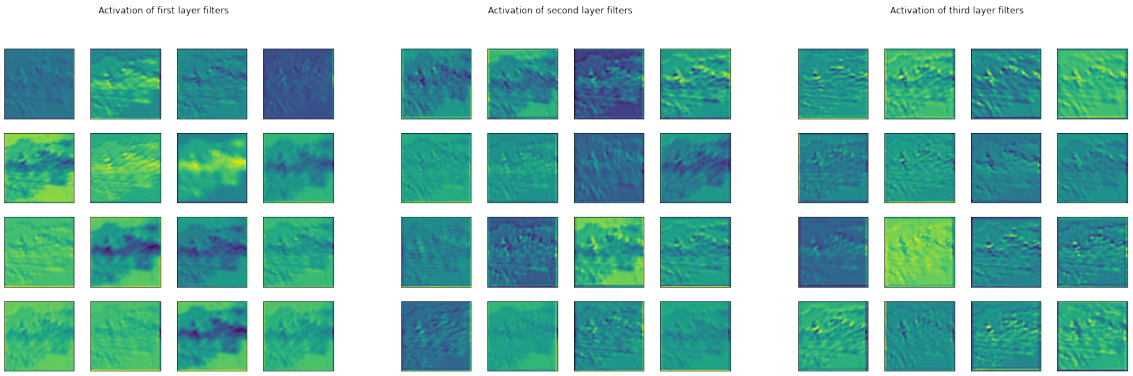
Overview of simplified model filter activations

I now overlay the activations from the CNNs on the three test images chosen earlier. Each row deals with a single image, and each column represents a convolutional layer. In theory we should see simpler features lower in the model - on the left hand side, and more complex features higher up in the model - on the right.

Activations for Case A



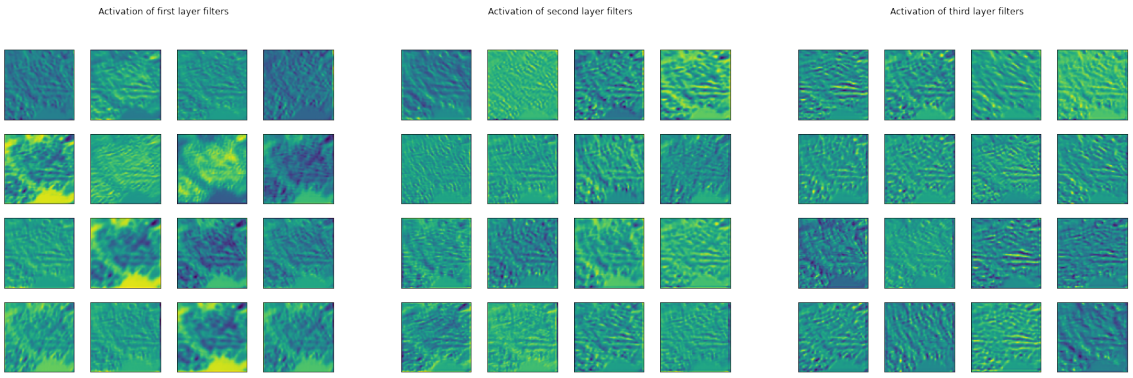
Activations for Case B



Activations for Case C

Brighter values indicate higher activations in those regions of the image. From inspecting the activations below, there appear to be a few general features:

- Some first level filters appear to detect sea, land, or edges



- For Case A, the second and third layers appear to build up a clear view of the coastline, which would indicate high relief
- It is harder to discern which, if any, filter are recognising cloud - it could be filter 7 which has fairly high activation across all cases
- For Case C, there does not appear to be any particularly highly-activated filter, which might represent a cloudy image overall

To summarise, it seems that this neural network is focusing on recognising an image with high relief between land and sea, and not separately recognising cloud. This makes sense because times when there is a high contrast between land and sea will be when the land is visible and highly reflective - so when the sun is high in the sky and there are few clouds. I will return to this in 6.

5.5 Computational and carbon cost

The motivation for this dissertation was to optimise the energy grid to reduce carbon emissions. But in doing so, I proposed a computational model which requires substantial computational resources, adding to energy demand and possibly increasing carbon emissions. Researchers in machine learning are beginning to explore the energy requirements and carbon intensity of large models. For example Strubell et al [45] find that the training and neural architecture search for a large transformer model could result in 284 metric tonnes of carbon emissions, compared to an individual on a 6-hour flight resulting in 0.9 metric tonnes. Does the increased forecast accuracy of superior models improve forecast accuracy enough to justify their energy requirements? In this section I estimate the energy requirements of the models from this dissertation. I will make several assumptions and revisit them in the limitations section below.

Each of the models was run on a Tesla T4 GPU available through Google Colab, which I will assume is in Northern Europe, the region closest to me, which has a carbon efficiency of 0.21 $kgCO_2eq/kWh$. Energy usage simulations were conducted using the Machine Learning Impact calculator presented in [28]. I take the two-hour prediction window and compare the ConvLSTM and 3D CNN. I assume that the predictions are generated for each of the 330 UK grid supply points, and are generated every hour, from 06:00 to 20:00, and for half of the year, giving 330 * (20-6) * 365 * 0.5 = 843,150 forecasts per year.

In terms of emissions reductions, Open Climate Fix estimates that better PV forecasts would reduce emissions in the UK by up to 100,000 tonnes per year [15]. Based on conversations with domain experts, they estimate the UK spinning gas reserve could be reduced by 40%, or 200

megawatts per day for approximately half of the year. Gas reserve produces approximate 1,000 tonnes of CO_2 per megawatt. Together this offers 200 MW of saving * 1/2 the year * 1,000 tonnes of CO_2, or 100,000 tonnes per year. In reality, National Grid ESO already has forecasts which beat persistence, but these taken in multiple inputs, and so for simplicity I compare results against a baseline.

It is hard to know exactly how much increased forecast accuracy would lead to reduced gas turbine usage, as this depends on the grid operators' confidence in the models. I present a range from where the increased accuracy relative to baseline translates into carbon savings either at a ratio of 1:10,000 to 1:10.

	CNN	**ConvLSTM**
Time to train model (s)	916.39	673.11
Time for inference, one forecast (s)	0.16	2.21
$\implies$ Total time for year (training + 843,150 * inference) (hrs)	37	517
Emissions generated (kg CO_2-equiv) from [28]	**0.54**	**7.6**
MAE (watts)	13.75%	12.52%
Improvement over persistence (watts)	6.57%	7.80%
Emissions averted (kg CO_2-equiv)	**657-657,000**	**780-780,000**
Net emissions averted (kg CO_2-equiv)	**656-656,999**	**772-779,992**

Table 5.2: Estimates of emissions generated and reduced

.

It seems that even under pessmistic assumptions, an improvement in the PV forecast could lead to at least a 100 times more emissions averted than generated.

Limitations of computational and carbon cost analysis

The above analysis has many limitations. From the emissions generated side, the energy production from the data centre can come from many different sources. I use a calculator provided by Lacoste et al [28], which takes available information about data centre energy supply. However, it is not clear when using Google Colab in which region the calculations are performed, and the energy mix will also vary throughout the day and year.

Perhaps there are even more limitations in terms of estimated benefit. I estimate benefit using approximate calculations from Open Climate Fix. In order to meaningfully reduce carbon emissions, a forecast PV model would not need to have a lower MSE and RMSE - it would need to convince the grid operators at ESO National Grid. Only if they were more confident in this model, and then reduced spinning gas turbines as a result, then can we argue that improved accuracy led to reduced emissions. So the relationship between increased accuracy of a predictive model and emissions averted is not likely to be linear.

In a 2021 paper published in *Nature*, researchers from Deepmind worked with expert meteorologists from the UK Met Office to develop a deep generative model to predict precipitation [36]. They found that the meteorologists were interested in models which led to predictions which looked better and represented possible weather conditions, not simply models with the lowest error measures. Grid operators may similarly be more interested in realistic PV predictions, but not necessarily the most accurate ones, which can then feed into their own decision-making.

Chapter 6

Discussion and conclusions

Chapter summary: I discuss the meaning and implications of the results. Weather prediction involves a mixture of deterministic (e.g. brightness) and stochastic (e.g. cloud motion) features. I consider CNNs and ConvLSTMs in relation to the task of solar PV nowcasting using satellite imagery (6.1), and explore the results from analysing prediction errors and the convolutional filters (6.2). The following section (6.3) looks at implications for machine learning and solar PV nowcasting. Finally, I consider directions for future research (6.4) and reflect on the project's achievements (6.5).

6.1 Discussion of main results

In my experiments, I found that there is moderate evidence that time series modelling in recurrent networks has higher predictive accuracy, especially over longer time periods. I also found that satellite images improve predictions, especially when using the recurrent LSTM architecture.

Why might LSTMs outperform CNNs on this specific task?

I will start with the comparison between LSTMs and CNNs. The difference in RNNs is that they pass the information back in to update their predictions, includes a temporal aspect absent from CNNs. This is sometimes described as the 'memory', and is based on a biological model of information processing, where the value in a time series is processed at each moment, but in combination with memories and context.

CNNs were developed for machine vision, and assume translation invariance. This means CNNs assume that it does not matter where in the data object a particular feature appears. This makes sense in vision, because the range of perspectives and distances between an object and the perceiver means we should still recognise a particular feature regardless of where it appears.

When CNNs are applied to a block of images, the model effectively flatten the entire network into one high-dimensional vector. This means the time dimension is treated in the same ways as any other dimension. But in RNNs, the sequence and ordering does matter. But in weather prediction, the time dimension will matter more because later images are likely to have more predictive power.

LSTMs are a step better than RNNs because they carry forward a richer thread of information than a single state vector. LSTMs can remember some information, but forget others.

As discussed in the literature review in Chapter 2, there is debate over whether LSTMs outperform CNNs, such as in Bai et al [2].

In our context, we found that the LSTMs had a more significant improvement over CNNs for longer time horizons. A potential explanation is that the LSTM is able to learn context vectors which say whether the day is getting brighter or darker, and these vectors are able to be remembered better and take more predictive importance.

Why might ConvLSTMs outperform 3D CNNs on this task?

I will now include the comparison with ConvLSTMs and 3D CNNs. We generally found that satellite imagery improved predictive accuracy, but the biggest differences were over longer time scales. Why might the ConvLSTMs outperform CNNs over longer time scales?

One reason could be that weather predictions images include both constant features and temporary aspects. For example, an important and constant feature might be a darker image overall, such as in the evening; and temporary aspect could be transient cloud pattern which might might build to provide cover, or might dissolve, leaving a clearer sky. The combination of states and updating in LSTMs and ConvLSTMs might provide further capacity to learn which features are constant and which ones are transient.

Our results suggest that ConvLSTMs outperform CNNs, despite argument such as in Bai et al [2] that CNNs can outperform LSTMs. A 3D convolutional neural network will start by treating all dimensions in the three-dimensional block as equally important, and will need to use some of its neurons to learn the time dependency. By including time explicitly as a dimension in time series, we provide additional information and so allow the neural network to learn features more useful for identifying PV yield.

Patterns in prediction errors - a gloomy model

From analysing prediction error, we see that the model consistently under-predicts PV yield, especially so in the early afternoon in summer. This could be caused by the model having insufficient training data for sunny days, and under uncertainty to revert towards the mean. This could be further explored by training different models on summer and winter, and seeing if the same effect persists. MSE equally penalises over- and under- predictions, and since the early morning and evening tend to have low or near-zero PV yield, there are many cases where predicting a low value is a safe bet. This could also be explored by altering the loss function to more heavily penalise under-predictions.

6.2 Interpretability

What is the model learning?

In terms of interpretability, I briefly examined a simpler model using only a CNN structure. The lack of activation for cloud patterns, and higher regions of activations for land, sea, and edges suggests that this model is learning to recognise a high-relief image of the land. This makes sense because a high constrast image of the land against the sea would imply a bright sun and minimal cloud cover. Perhaps the model is not learning to recognise cloud dynamics, but instead simply recognising when the land and the sea are clearest.

6.3 Implications

Implications for machine learning

The main finding in this dissertation was that explicitly modelling data as a time series improved predictive accuracy. Rich Sutton [46] argued that the 'bitter lesson' of AI research is that general methods that leverage computation are ultimately the most effective and accurate. Sutton claims that general approaches are better because of the 'arbitrary, intrinsically-complex, outside world' but he does not rule out the use of time as a dimension. In weather forecasting, time seems to be a particularly important dimension - with a higher degree of predictive importance in solar PV nowcasting than finding patterns in other areas.

Since 2017, the transformer architecture [47] has become a dominant approach in computer vision. The transformer architecture is general and defines a broad hypothesis space in terms of time-dependency. We found that explicit representation of time series improved predictions. This suggests it may be possible to improve on general architectures by leveraging recurrent relations.

From examining the convolutional layer activations for a simpler model, I found that the model appeared to be recognising the contrast between the land and the sea, and not necessarily learning about cloud dynamics. This is a case of 'getting what you ask for, not what you want', a perennial issue in machine learning. Only by examining the activations was this apparent. If a model like this was deployed on a different geographical region with no coast line, its predictive accuracy would likely be far lower. This highlights the importance of interpreting how machine learning models derive their predictions, and ensuring that the datasets used in training are representative of real data in deployment.

Implications for solar PV nowcasting

This dissertation focused on predicting solar PV in the UK, and found that several relatively simple model was able to outperform a persistence baseline. In fact models which only took past PV readings as input had competitive performance up to two hours. This suggests that relatively computationally cheap models could be useful in anticipating solar PV dynamics, and highlights the use of having timely PV data made easily available.

I also found that even with relatively pessimistic assumptions, the benefits of reduced carbon emissions from more accurate solar PV readings outweight the associated carbon emissions with running the models, on a scale of at least 1:100. However, as models become larger and improvements in accuracy become more marginal, there may be false economies. In order to reduce environmental impact, larger models should try to run as efficiently as possible, and investigate the energy sources behind their data centres.

Inequalities in access to data

One limitation is that solar PV nowcasting research disproportionately focuses on a small number of wealthier countries. In the meta-analysis of papers on solar PV nowcasting, Martins et al [31], the authors also find that studies using data from the US outnumber any other country, and only 3 of 45 papers are identified as using datasets from non-OECD countries.

Energy inequalities

Moreover, the UK has relatively low potential for energy solar PV than other countries. The chart below, based on data from the World Bank [3] shows the potential energy achievable through solar PV against the proportion of per-capita energy demand met through solar energy. The UK is highlighted in green, as one of the countries with the lowest potential for solar energy due to its latitude and geography (on the y axis), but among the highest proportion of solar PV as a proportion of energy demand (x axis).

Figure 6.1: A graph showing solar capacity against solar potential for individual countries - the UK has among the worst potential for PV but one of the highest portions of its energy provided by solar PV.

In the famous *Sustainable Energy - without the hot air*, the late David MacKay argued that solar PV could be a way to reduce your personal average electricity consumption, but that in the UK the combination of high population density and high latitude would make it hard to deploy enough solar to put a get enough energy to put a significant dent in total energy consumption.

"*It would quite probably be better to put the panels in a two-fold sunnier country and send some of the energy home by power lines* ([30], p. 41)"

In his book published in 2009, MacKay mentions the DESERTEC plan [7] which uses larger concentrating solar power, rather than solar panels, in sunnier Mediterranean countries, and then delivers the energy using long-distance high-voltage direct-current transmission lines. This project appears to have stalled in the subsequent fourteen years, but the point remains that the UK might not be the best country to deploy large amounts of solar energy.

If further research aims to reduce carbon emissions, it might be fruitful to focus on relatively under-studied countries with a higher potential for PV, where the scope for integrating more energy from solar PV into the electrical grid is greater.

6.4 Future work

There are multiple possible directions in which these results could be improved. The most straightforward would be to repeat the experiment with larger and different datasets. I focused on a subset of the UK, but a model could be trained on many randomly selected points. Open Climate Fix divide the UK into approximately 330 sections, corresponding to the grid supply points (GSPs) used by National Grid. In addition, this model has only looked at the UK but it would be useful to perform experiments on other countries with lots of installed solar PV capacity, and with a greater potential for PV, such as Spain, Morocco, and Egypt. Even the data within the UK could be significantly improved, as this model used 1300 solar PV stations, when there are at least 260,000 solar PV installations where the best available datasets estimate 86% coverage [44].

In terms of more advanced models, Xu et al [48] develop a combination of a GAN and an LSTM for prediction of cloud images. Since around 2017, attention-based models have been used for sequence prediction tasks [1] and would provide another way to examine dependencies in time series data, though as mentioned earlier, it might be helpful to include explicit representations of time series information to reduce the search space.

A different approach is to develop a model to predict a sequence of frames, and then use a second model to infer PV yield from these images. Jack Kelly from Open Climate Fix mentioned this as an area of potential future work, with the reason that there is larger dataset for the first task - all satellite images - and that the main source of uncertainties in forecasts is cloud dynamics. Therefore segmenting the task into separately predicting cloud dynamics from PV yield could be a promising area for further research.

6.5 Achievements

In this section I assess the goals set out at the beginning of the project, in section 1.3.

Goal	**Status**	**Progress**
Specify a task	✓	Identified solar PV forecasting as a computational task building on concepts covered in my course at UCL
Data collection	✓	Identified two real-world datasets used to train models used in industry
Preprocessing	✓	Examined and selected subsets of the data, and split into valid test/train/validation sets, then aligned, normalised and cleaned data
Model selection	✓	Learned how ANNs, CNNs, RNNs, and ConvLSTMs are used, and what approaches are used by OCF, then implemented my own versions of these models
Results evaluation	✓	Trained models on datasets, tuned hyperparameters, and critically assessed results in relation to findings in the literature and at OCF

Table 6.1: Progress against project goals set in section 1.3

.

Appendix A

Code

All of the code from this dissertation is available at `https://drive.google.com/drive/folders/1IgZpWxcXFueEvTHJptnb-CgdJP6xr_ld`. The contents of the folder are as follows:

- `data_pipeline`: pipeline and preprocessing of satellite and PV data
- `model_evaluation`: comparison of performance across models and time periods
- `model_files`: saved model files after training
- `unused_mode_notebooks`: as described in Chapter 1, I followed an Agile approach and experimented - this folder contains each increment of the models as my understanding developed over the project
- `Models 20, 40, 41, 42, 43`: models as described in experimental design
- Multiple `Report` files, used to generate predictions, calculations, and figures shown in this report

Bibliography

[1] Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. "Neural Machine Translation by Jointly Learning to Align and Translate". In: *3rd International Conference on Learning Representations, ICLR 2015,* 2015. URL: http://arxiv.org/abs/1409.0473.

[2] Shaojie Bai, J. Zico Kolter, and Vladlen Koltun. *An Empirical Evaluation of Generic Convolutional and Recurrent Networks for Sequence Modeling.* URL: https://arxiv.org/abs/1803.01271.

[3] World Bank. *Solar Photovoltaic Power Potential by Country.* 2020. URL: https://www.worldbank.org/en/topic/energy/publication/solar-photovoltaic-power-potential-by-country.

[4] Akansha Singh Bansal, Trapit Bansal, and David Irwin. "A moment in the sun: solar nowcasting from multispectral satellite data using self-supervised learning". In: *e-Energy '22: Proceedings of the Thirteenth ACM International Conference on Future Energy Systems* (2022), pp. 251–262. DOI: https://doi.org/10.1145/3538637.3538854.

[5] Kyle Barron. *suncalc.* URL: https://pypi.org/project/suncalc/.

[6] Francois Chollet. *Deep Learning with Python (Second Edition).* Manning, 2021.

[7] *DESERTEC Foundation.* 2022. URL: https://www.desertec.org/.

[8] Natioal Grid ESO. *Future Energy Scenarios: July 2022.* 2022. URL: https://www.nationalgrideso.com/future-energy/future-energy-scenarios.

[9] National Grid ESO. *Future of GB's Electricity National Control Centre.* 2019. URL: https://www.nationalgrideso.com/news/future-gbs-electricity-national-control-centre.

[10] National Grid ESO. *The Road to Zero Carbon.* 2022. URL: https://www.nationalgrideso.com/future-energy/our-progress/road-zero-carbon/report.

[11] National Grid ESO. *What is Frequency?* 2022. URL: https://www.nationalgrideso.com/electricity-explained/how-do-we-balance-grid/what-frequency.

[12] EUMETSAT. *Meteosat Second Generation.* 2022. URL: https://www.eumetsat.int/meteosat-second-generation.

[13] EUMETSAT. *SEVIRI.* 2022. URL: https://www.eumetsat.int/seviri.

[14] Open Climate Fix. *EUMETSAT Dataset.* 2022. URL: https://huggingface.co/datasets/openclimatefix.

[15] Open Climate Fix. *Forecasting.* 2022. URL: https://www.openclimatefix.org/projects/forecasting/.

[16] Yarin Gal and Zoubin Ghahramani. "Dropout as a Bayesian Approximation: Representing Model Uncertainty in Deep Learning". In: *Proceedings of the 33rd International Conference on International Conference on Machine Learning - Volume 48*. ICML'16. New York, NY, USA: JMLR.org, 2016, pp. 1050–1059.

[17] Aurelien Geron. *Hands-On Machine Learning with Scikit-Learn, Keras, and Tensorflow (Second Edition)*. 2019.

[18] F.A. Gers and J. Schmidhuber. "Recurrent nets that time and count". In: *Proceedings of the IEEE-INNS-ENNS International Joint Conference on Neural Networks. IJCNN 2000. Neural Computing: New Challenges and Perspectives for the New Millennium*. Vol. 3. 2000, 189–194 vol.3. DOI: 10.1109/IJCNN.2000.861302.

[19] Google. *Google Colaboratory*. 2022. URL: https://colab.research.google.com/.

[20] Alex Graves. *Generating Sequences With Recurrent Neural Networks*. Aug. 2013. URL: https://arxiv.org/abs/1308.0850.

[21] Sepp Hochreiter and Jürgen Schmidhuber. "Long Short-Term Memory". In: *Neural Computation* 9.8 (1997), pp. 1735–1780.

[22] IPCC. *AR6 Working Group II: Impacts, Adaption, and Vulnerability - Summary for Policymakers (SPM)*. 2022. URL: https://www.ipcc.ch/report/ar6/wg2/.

[23] IPCC. *AR6 Working Group III Report: Mitigation of Climate Change - Summary for Policymakers (SPM)*. 2022. URL: https://www.ipcc.ch/report/sixth-assessment-report-working-group-3/.

[24] Diederik P. Kingma and Jimmy Ba. "Adam: A Method for Stochastic Optimization". In: *ICLR (Poster)*. 2015. URL: http://arxiv.org/abs/1412.6980.

[25] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. "ImageNet Classification with Deep Convolutional Neural Networks". In: *Advances in Neural Information Processing Systems*. Ed. by F. Pereira et al. Vol. 25. Curran Associates, Inc., 2012. URL: https://proceedings.neurips.cc/paper/2012/file/c399862d3b9d6b76c8436e924a68c45b-Paper.pdf.

[26] Y. Lecun et al. "Gradient-based learning applied to document recognition". In: *Proceedings of the IEEE* 86.11 (1998), pp. 2278–2324. DOI: 10.1109/5.726791.

[27] Yann LeCun, Yoshua Bengio, and Geoffrey Hinton. "Deep learning". In: *Nature* 521.7553 (2015), pp. 436–444. ISSN: 0028-0836.

[28] Sasha Luccioni et al. "Quantifying the Carbon Emissions of Machine Learning". In: (2019). URL: https://www.climatechange.ai/papers/neurips2019/22.

[29] Andrew L. Maas, Awni Y. Hannun, and Andrew Y. Ng. "Rectifier nonlinearities improve neural network acoustic models". In: *in ICML Workshop on Deep Learning for Audio, Speech and Language Processing*. 2013.

[30] David MacKay. *Sustainable Energy - without the hot air*. 2009. URL: https://www.withouthotair.com/.

[31] Bruno Juncklaus Martins et al. "Systematic review of nowcasting approaches for solar energy production based upon ground-based cloud imaging". In: *Solar Energy Advances* (2022). DOI: https://doi.org/10.1016/j.seja.2022.100019.

[32] Warren S. McCulloch and Walter Pitts. "A logical calculus of the ideas immanent in nervous activity". In: *Bulletin of Mathematical Biophysics* (1943), pp. 115–133. DOI: https://doi.org/10.1007/BF02478259.

[33] Michael A Nielsen. *Neural Networks and Deep Learning*. Determination Press. URL: http://neuralnetworksanddeeplearning.com/.

[34] Christopher Olah. *Understanding LSTM Networks*. 2015. URL: http://colah.github.io/posts/2015-08-Understanding-LSTMs/.

[35] Rachel Prudden et al. *A review of radar-based nowcasting of precipitation and applicable machine learning techniques*. 2020. URL: https://arxiv.org/abs/2005.04988.

[36] Suman Ravuri et al. "Skilful precipitation nowcasting using deep generative models of radar". In: *Nature* 597 (2021).

[37] Matt Reynolds. *Bad weather forecasts are a climate crisis disaster*. 2021. URL: https://www.wired.co.uk/article/solar-weather-forecasting.

[38] Frank Rosenblatt. "The perceptron: A probabilistic model for information storage and organization in the brain". In: 65 (1958), pp. 386–408. DOI: https://doi.org/10.1037/h0042519.

[39] David E. Rumelhart, Geoffrey E. Hinton, and Ronald J. Williams. "Learning representations by back-propagating errors". In: *Nature* 323 (1986), pp. 533–536.

[40] Martin Schultz et al. "Can deep learning beat numerical weather prediction?" In: *Philosophical Transactions of The Royal Society A Mathematical Physical and Engineering Sciences* 379 (Feb. 2021). DOI: 10.1098/rsta.2020.0097.

[41] Xingjian Shi et al. "Convolutional LSTM Network: A Machine Learning Approach for Precipitation Nowcasting". In: *Advances in Neural Information Processing Systems*. Ed. by C. Cortes et al. Vol. 28. Curran Associates, Inc., 2015. URL: https://proceedings.neurips.cc/paper/2015/file/07563a3fe3bbe7e3ba84431ad9d055af-Paper.pdf.

[42] Ian Sommerville. *Software Engineering*. 9th ed. 2010.

[43] Liam Stoker. *'Unprecedented' events send UK power market into negative pricing for six hours straight*. 2019. URL: https://www.current-news.co.uk/news/unprecedented-events-send-uk-power-market-to-negative-pricing-for-six-hours-straight.

[44] Dan Stowell et al. "A harmonised, high-coverage, open dataset of solar photovoltaic installations in the UK". In: *Scientific Data* 7 (2020). DOI: https://doi.org/10.1038/s41597-020-00739-0.

[45] Emma Strubell, Ananya Ganesh, and Andrew McCallum. "Energy and Policy Considerations for Deep Learning in NLP". In: (2019). DOI: 10.18653/v1/P19-1355.

[46] Rich Sutton. *The Bitter Lesson*. 2019. URL: http://www.incompleteideas.net/IncIdeas/BitterLesson.html.

[47] Ashish Vaswani et al. "Attention is All you Need". In: *Advances in Neural Information Processing Systems*. Ed. by I. Guyon et al. Vol. 30. Curran Associates, Inc., 2017. URL: https://proceedings.neurips.cc/paper/2017/file/3f5ee243547dee91fbd053c1c4a845aa-Paper.pdf.

[48] Zhan Xu et al. "Satellite Image Prediction Relying on GAN and LSTM Neural Networks". In: *ICC 2019 - 2019 IEEE International Conference on Communications (ICC)*. 2019, pp. 1–6. DOI: 10.1109/ICC.2019.8761462.