Final Year Project Cover Sheet
Department of Civil and Environmental Engineering

Name	Yen Hann YOO
Module	**CIVE97003 Individual Research Project**
Project Title	Short-Term Rainfall Cell Forecasting using Machine Learning Techniques
Supervisor	Dr. Christian Onof
Deadline	4th June 2021

DECLARATION

I certify that I have read the definition of plagiarism given overleaf, and that the work submitted for this coursework assignment is my own work, except where specifically indicated otherwise. In signing this document, I agree that this work may be submitted to an electronic plagiarism test at any time and I will provide a further version of this work in an appropriate format when requested:

Signature: Yen Hann Yoo **Date:** 4/6/2021

Note: Until an assignment carries this completed front page it will not be accepted for marking. If the front page is absent, the delay in getting it added may result in a penalty for late submission.

PLAGIARISM

You are reminded that all work submitted as part of the requirements for any examination (including coursework) of Imperial College must be expressed in your own writing and incorporate your own ideas and judgements.

Plagiarism, that is the presentation of another person's thoughts or words as though they are your own, must be avoided with particular care in coursework, essays and reports written in your own time. Note that you are encouraged to read and criticise the work of others as much as possible. You are expected to incorporate this in your thinking and in your coursework and assessments. But you must acknowledge and label your sources.

Direct quotations from the published or unpublished work of others, from the internet, or from any other source must always be clearly identified as such. A full reference to their source must be provided in the proper form and quotation marks used. Remember that a series of short quotations from several different sources, if not clearly identified as such, constitutes plagiarism just as much as a single unacknowledged long quotation from a single source. Equally if you summarise another person's ideas, judgements, figures, diagrams or software, you must refer to that person in your text, and include the work referred to in your bibliography and/or reference list. Departments are able to give advice about the appropriate use and correct acknowledgement of other sources in your own work.

The direct and unacknowledged repetition of your own work, which has already been submitted for assessment, can constitute self-plagiarism. Where group work is submitted, this should be presented in a way approved by your department. You should therefore consult your tutor or course director if you are in any doubt about what is permissible. You should be aware that you have a collective responsibility for the integrity of group work submitted for assessment.

The use of the work of another student, past or present, constitutes plagiarism. Where work is used without the consent of that student, this will normally be regarded as a major offence of plagiarism.

Failure to observe any of these rules may result in an allegation of cheating. Cases of suspected plagiarism will be dealt with under the College's Cheating Offences Policy and Procedures and may result in a penalty being taken against any student found guilty of plagiarism.

Short-Term Rainfall Cell Forecasting using Machine Learning Techniques

Yen Hann YOO

Department of Civil and Environmental Engineering
Imperial College London

Abstract

Two Convolutional Neural Networks were adapted for rainfall forecasting in Birmingham to predict future rainfall images and identify high rainfall pixels. The first model is trained on rainfall images only. The second model is trained on climatological features and rainfall images. The effectiveness of both models in predicting high rainfall pixels were assessed. These models are evaluated quantitatively and qualitatively against a test set of approximately 3000 to 4000 rainfall images. The first model exhibited a hit rate of approximately 20% whereas the second model exhibited a hit rate of < 1%. These results indicated that the first model is better at identifying high rainfall pixels and hence, better predicted areas prone to flooding than the second model. However, it cannot be concluded with certainty that this is indeed the case because assumptions were made to incorporate the climate data into the model due to its higher temporal resolution in comparison to the rainfall images. This paper also compares the two models and provides suggestions on how to improve both models' predictive capabilities.

1. Introduction

1.1 Background

Prolonged storm events of high rainfall intensities have the potential to induce destructive floods. Jackson (2014) states that such storm events results in oversaturated ground that leads to increased surface runoff and therefore flooding. Furthermore, increased areas of impervious paved ground due to urbanisation exacerbates the likelihood and destructiveness of these rainfall induced floods. Moreover, the warming effects of climate change in recent years have resulted in increased rainfall intensities. According to Massam (2020), the Clausius-Clapeyron relationship dictates that atmospheric water content increases by 6 – 7% for every degree Celsius increase in surface temperature. This positive correlation is also supported by recent evidence indicating that the rain volumes from extremely wet days has increased by 17% when comparing periods between 1961 – 1990 and 2008 – 2017 (UK Met Office, 2021) with 19 of the warmest years recorded in history occurring since 2000 (NASA, 2021)

These changing dynamics of storm events and the destructive floods they can bring about therefore necessitates the development of reliable rainfall forecasting models. These models' outputs can be fed into flood prediction models to enable systematic dissemination of warnings to authorities and the populace, thereby minimising damage, and loss of life. Today, the intensities and movements of rainfall systems are forecasted through two types of models – physical and data-driven models. Physical models such as Numerical Weather Prediction (NWP) rely on the physical processes of weather systems such as heat flux, atmospheric turbulence, etc. by employing Partial Differential Equations (PDEs) describing fluid flow. These PDEs are subsequently discretised, processed, and solved by computers (NOAA, 2021).

Conversely, data-driven models do not draw upon the physics describing rainfall events but attempts to learn statistical relationships between the inputs and outputs. In rainfall forecasting, inputs would constitute features such as humidity, temperature, etc. whilst the output would be the rainfall intensities. By quantifying these relationships, these models can predict outputs with different input values. Data-driven models consist of statistical models and Machine Learning (ML) models. Traditional statistical methods used in forecasting are the Auto-Regressive Integrated Moving Average (ARIMA) models, which uses a specified number of lagged observations of a time series to forecast future observations. A weight is associated with each past observation and the weights will vary based on how recent each observation is (Malik, 2018).

However, ARIMA models suffers from drawbacks such as the inability to predict sudden changes and turning points in the series (Zhai, 2005). These models also assume variables to be Gaussian whereas rainfall depths often exhibit skewed distributions and strong intermittency. These disadvantages along with the popularity of Artificial Intelligence, have fuelled the proliferation of ML models. Today, most ML problems can be divided into two groups: supervised learning and unsupervised learning (Garbade, 2018). Supervised learning encompasses problems where for every X, Y is also known. Therefore, the goal of supervised learning is to determine a function that most accurately maps the inputs to their outputs. The two main areas of supervised learning problems are classification and regression, where the former is categorical and the latter numerical (Garbade, 2018).

Conversely, in unsupervised learning, there is no prior knowledge of the outputs. Hence, the goal of unsupervised learning is to infer the natural structure and relationship between samples within the dataset (Soni, 2018). The two main areas of unsupervised learning problems are clustering and association (Garbade, 2018). In rainfall forecasting problems, the independent and dependent variables are usually known quantities and hence, rainfall forecasting can be regarded as a regression problem as the output is typically a continuous quantity. Examples of ML models utilised in regression problems include Neural Networks and Support Vector Regression (SVR).

This paper aims to investigate the effectiveness of implementing a subset of Neural Networks, known as Convolutional Neural Networks (CNNs), in forecasting future rainfall images for the city of Birmingham to identify areas susceptible to high rainfall. This research can be divided into two phases. The first phase will investigate the efficacy of these CNNs when trained only on a database of historical rainfall images. The second phase incorporates climatological data with historical rainfall images and compares the results of both investigations. The novelty of this research lies in the second phase whereby the incorporation of climatological data in addition to rainfall images has never been investigated before.

1.2 Study Area

Birmingham is the second largest city in the United Kingdom with 1.1 million inhabitants and is susceptible to flooding due to topographical factors and urbanisation (Birmingham City Council & Atkins, 2012). Since 1910, 9 of the 17 record-breaking rainfall months have been registered after 2000. These storm events are associated with high rainfall intensities and have been known to induce floods. The frequency of these events is also expected to increase with climate change. Birmingham typically experiences 660mm of rainfall annually and the wettest and driest months correspond to August/December and July respectively (ClimaTemps, 2017).

Keeling (2005) states that Birmingham frequently experiences storms as winds from the south are forced up the Birmingham Plateau, which subsequently forms large storm clouds. Due to the city's urbanised setting, large volumes of rainwater from storm events cannot infiltrate into the ground or sewers quick enough, which leads to flooding. Birmingham's sewer network is also outdated as it was built in the 18th century and thus, are inundated during periods of high rainfall as they were designed to handle smaller volumes of rainwater (Birmingham City Council & WSP, 2017).

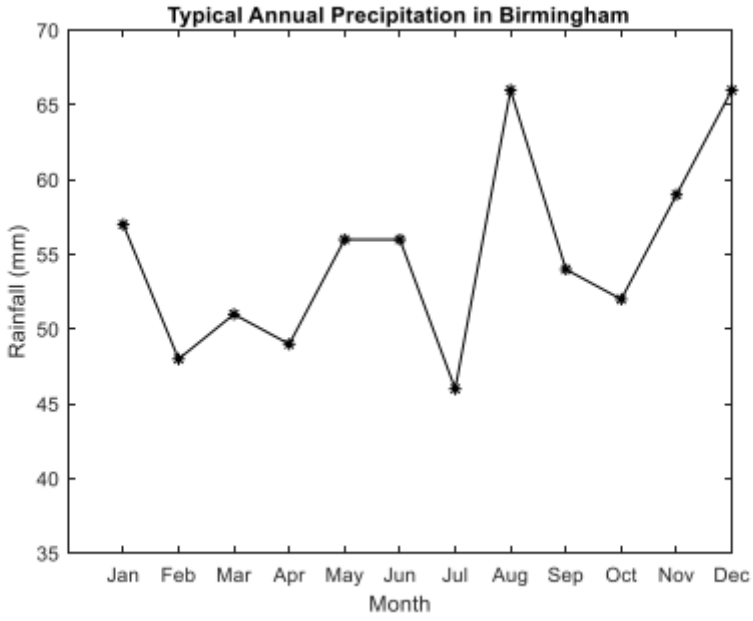


Figure 1 | Precipitation graph for Birmingham (ClimaTemps, 2017)

1.3 Aims and Objectives

The aim of this project is to develop an extreme rainfall forecasting model using CNNs for the city of Birmingham, United Kingdom by first, using only rainfall radar data and second, by adding climatological features as independent variables. Subsequently, the results and effectiveness of both methods will be evaluated.

The objectives applicable to the aims of this project are the following:

- Literature review on rainfall forecasting models.
- Pre-processing of radar and climatological data.
- Train CNNs on rainfall radar data only to forecast future rainfall images and identify areas susceptible to high rainfall.
- Evaluate the above models by comparing their forecasts with the test set.
- Train CNNs on both rainfall radar data and climatological data to forecast future rainfall images and identify areas susceptible to high rainfall.
- Evaluate the above models by comparing their forecasts on the test set with CNNs trained on rainfall images only.

2. Literature Review

2.1 Data Pre-Processing

Prior to developing any ML model, the input data must be pre-processed, cleaned and if necessary, reshaped into an appropriate format. This commonly involves imputing missing values with averages, removal of outliers and feature scaling. These steps are imperative in ensuring the robustness and accuracy of the chosen ML model.

2.1.1 Feature Scaling

Feature scaling ensures inputs and outputs are on a similar scale. This enables optimization algorithms to converge faster during the training process (Valkov, 2019) and minimizes the likelihood of erroneous predictions. A common feature scaling technique is feature normalisation - rescaling inputs to a normalised range (Bhandari, 2020). This is typically achieved using min-max scaling, whereby all values are transformed to range between 0 and 1. For any feature X_j, Equation (1) describes its normalised form X^N_j.

$$X_j^N = \frac{X_j - min(X_j)}{\max(X_j) - min(X_j)} \quad (1)$$

2.2 Overview of Machine Learning Models in Rainfall Forecasting Literature

In a rainfall forecasting investigation conducted by Liu et. al. (2019), the input data consisted of tabulated climate features such as the humidity, dew temperature, etc. The aim was to forecast rainfall intensities at a specific location under different climate conditions. These factors therefore appropriated the use of a Feed Forward Neural Network with a single output neuron yielding the rainfall intensity. Conversely, in a different rainfall forecasting investigation conducted by Ayzel et. al. (2019), the input data consisted of historical rainfall images and the aim was to forecast future rainfall images. As such, CNNs were utilised due to their popularity in image-related problems.

From these examples, both studies had differing data characteristics (i.e., tabulated climate features vs rainfall radar images), which mandated two different models even though the same dependent variable (rainfall intensities) was computed. Therefore, factors pertaining to the nature of the input and output data, training time, etc., are also vital considerations during the model selection stage. Hence, there is no 'best' model for rainfall forecasting but rather, the most appropriate model is dependent on the specifics of each problem.

Nevertheless, Neural Networks will be investigated in this project for two reasons. Firstly, Neural Networks are widely regarded as one of the most popular ML models (Anand, 2020) as they consistently outperform several ML models in most problems due its universality (Danka, 2020). Effectively, this can be interpreted as Neural Networks being capable of approximating almost any function mapping input to output, regardless of complexity and the number of inputs and outputs (Sahay, 2020). Secondly, several variants of Neural Networks exist such as RNNs (Recurrent Neural Networks) and CNNs. This means that Neural Networks can be adapted to almost any problem, allowing for enormous flexibility and versatility.

2.3 Artificial Neural Networks

Artificial Neural Networks (ANNs) are designed to reflect the behaviour of the human brain by mimicking how biological neurons signal to one another (IBM, 2020). The most common and basic type of ANNs are Feed-Forward Neural Networks (FNNs). FNNs are comprised of three types of layers – the input, hidden, and output layer, with each node connected to every node in the successive layer by links with associated weights and thresholds.

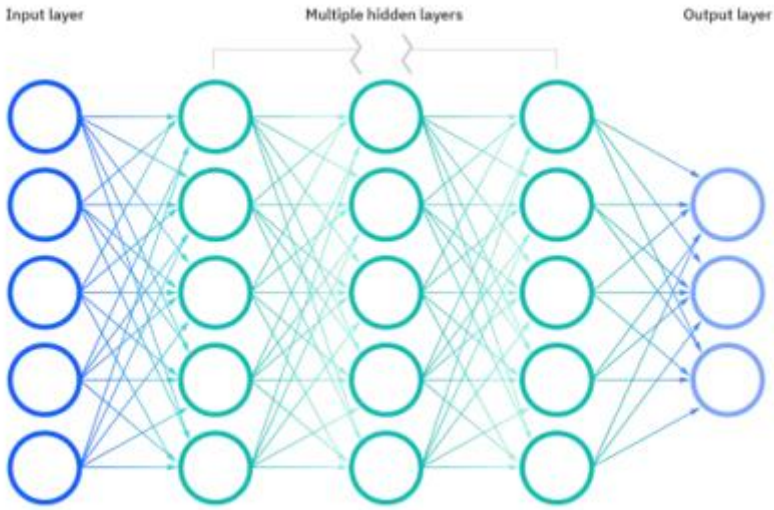


Figure 2 | Feed-Forward Neural Network (IBM, 2020)

Each node houses an activation function that transforms the nodal input into an output. If the output of any individual node exceeds the threshold value, that node is then 'activated,' thereby transmitting data to the successive layer and vice versa. In FNNs, a bias term is added to the weighted sum of nodal inputs. The bias term is represented by an additional neuron of a constant value or vector to offset the result of the activation function towards either the positive or negative region – analogous to the intercept term in y = mx + c (Malik, 2019).

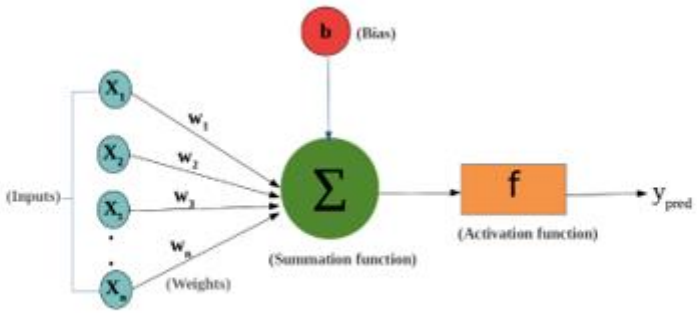


Figure 3 | Schematic representation of a neuron in FNNs (Ganesh, 2020)

Today, ANNs are employed in a multitude of applications such as forecasting and image recognition. Consequently, variations of ANNs have been specifically developed for different domains. For example, RNNs consist of cyclical connections between input and output neurons and are typically used in time series forecasting. On the other hand, CNNs are primarily used in image recognition. However, some research has been conducted towards utilising CNNs for image-to-image forecasting - predicting succeeding images in a sequence. Since rainfall images are a sequence of images at preceding timesteps, and the aim is to forecast future rainfall images, this problem therefore pertains to an image-to-image forecasting problem and appropriates the implementation of CNNs.

2.4 Convolutional Neural Networks

CNNs are analogous to the connectivity pattern of neurons in the human brain and are inspired by the organization of the visual cortex. CNNs receive images, assigns learnable weights/parameters during training to various features of the image (e.g., vertical edges, colour boundaries, textures, etc.) and subsequently differentiates one from the other (Saha, 2018). The architecture consists of a feature learning (encoding) stage and output (decoding) stage. In the encoding stage, images are reduced to a form that is easier to process whilst extracting the important features. Conversely, in the decoding stage, extracted information from the encoding stage is translated into a result that best classifies or best predicts the intended output. These stages are achieved through a combination of convolutional, pooling and fully connected layers.

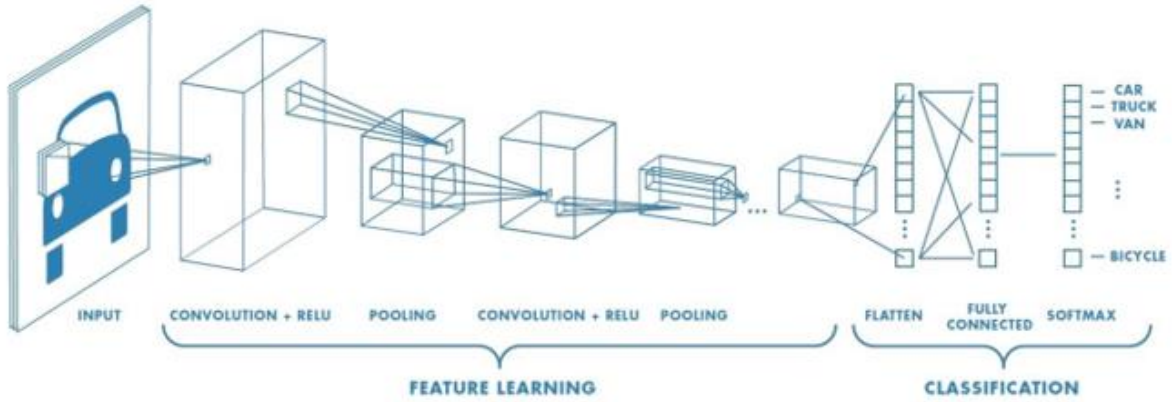


Figure 4 | Example of a CNNs architecture in image classification (Saha, 2018)

CNNs possess distinct advantages over FNNs in image-based problems. In these problems, images are described by pixel intensities stored in matrices. In FNNs, these matrices are 'flattened' into a vector prior to inputting into FNNs, which can yield satisfactory accuracies for basic black and white images (Saha, 2018). However, FNNs fail to capture the spatial and temporal dependencies between pixels in more complex (RGB) images, which CNNs circumvent through the application of relevant filters. Furthermore, CNNs can handle 3D inputs and do not require 'flattening,' and hence, are trained to understand the sophistication of an image dataset better (Saha, 2018).

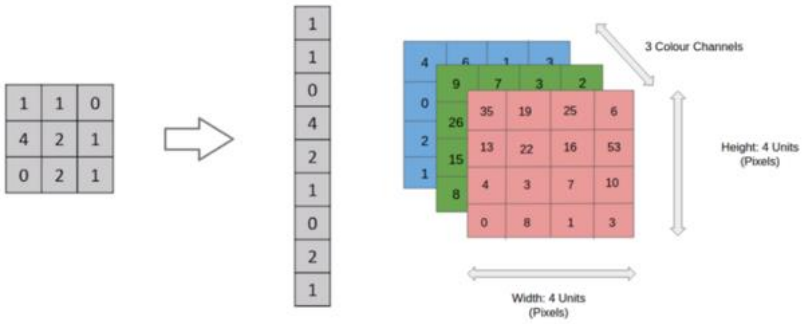


Figure 5 | Image inputs into FNNs (left) and CNNs (right) (Saha, 2018)

2.4.1 Layers

CNNs are comprised of three types of layers: *Convolutional*, *Pooling* and *Fully Connected* layers. The exact details on how these layers process input data is shown in Appendix A.

2.4.2 Padding

Padding refers to artificially increasing the height and width of input matrices by adding numbers around it. This enables preservation of the input dimensions as it passes through convolutional and pooling layers. If no padding is used, the input dimensions gradually reduce. Additionally, they also help to incorporate information from the input's corners - corner points are only convolved once without padding. Hence, less information is extracted from these points. The two most common types of padding are 'same' and 'valid' padding. Valid padding refers to no padding. Therefore, a layer's output is of smaller dimensions than its input. Same padding corresponds to padding where a layer's output is of equal size to its input. When same padding is applied, zeroes are typically padded to the input.

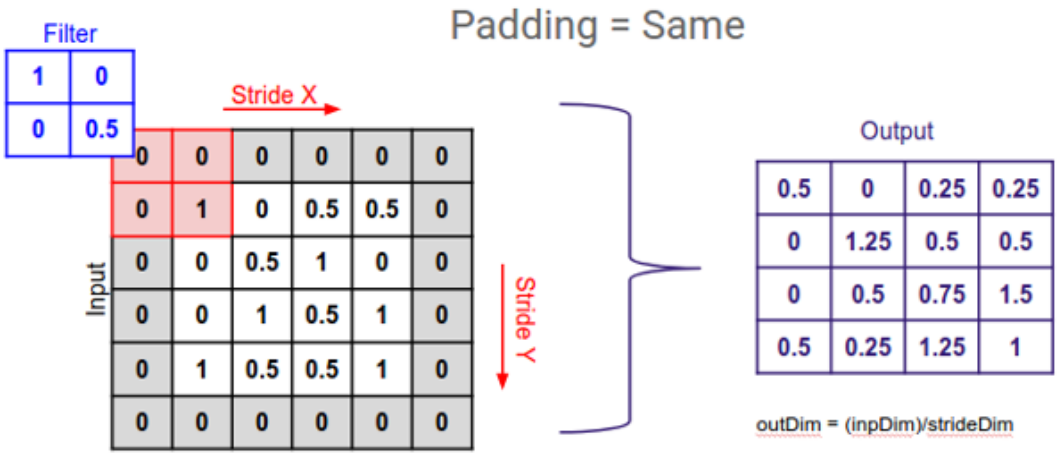


Figure 6 | Example of same padding (Perera, 2018)

2.4.3 Training Process

The training process refers to the 'learning' and optimization of weights and biases by iterating through the entire training dataset. With each subsequent iteration, the weights and biases are updated towards their optimal value using a chosen optimiser (training algorithm). Associated with each training process is the loss function, which is a mathematical function measuring the deviation between the true and predicted outputs.

Hence, the optimisation objective of the training process is to minimize this loss function and therefore determine the corresponding weights. Training CNNs typically consists of a forward propagation phase and backpropagation phase. In the forward propagation phase, all samples in the training dataset are passed through the network to generate predicted outputs, enabling the loss function to be calculated. Each layer will also retain any data (e.g., inputs, intermediate values, etc.) that will be used in backpropagation (Zhou, 2019). Therefore, forward propagation precedes backpropagation.

In the backpropagation phase, each layer will receive the derivative of the loss function with respect to the layer's outputs and returns the derivative of the loss function with respect to the layer's inputs (Zhou, 2019). The training process of CNNs can be summarised as follows:

1. Initialise the weights to arbitrary values and feed the training dataset through the network to obtain predictions

2. Calculate the loss (difference between true and predicted values) using the loss function

3. Calculate the gradients of the loss function with respect to the network's weights and biases for each layer

4. Update the parameters and biases using the gradients computed in the step 3 using a chosen optimizer/training algorithm to reduce the loss.

5. Repeat steps 1 to 4 until one epoch is completed.

   Epoch is a hyperparameter denoting the number of times the entire training dataset is forward propagated through the network from input to output.

6. Repeat steps 1 to 5 for as many epochs required to achieve the minimum loss.

3. Methodology

3.1 Overview

There are three main methodology sections: pre-processing, development of CNNs trained on rainfall images only and formulation of a CNN trained on rainfall images and climatological variables.

3.2 Pre-Processing

3.2.1 Rainfall Image Data

The rainfall image dataset consists of 14 datafiles containing a sequence of 105,120 rainfall images with a temporal resolution of 5-minutes and a spatial resolution of 1km. Each datafile corresponds to the rainfall images for a given year between 2005 - 2018. Each 5-minute rainfall image covers an area of 110km x 110km. The spatial extent of each rainfall image extends from (350,000, 230,000) to (459,000, 339,000) in Ordinance Survey Coordinates. However, the region of interest in Birmingham is a 34km x 30km area extending from (390,000, 270,000) to (419,000, 303,000).

3.2.2 Storm Event Inventory

Records of historical storm events were also provided, which contains information on 157 unique storm events scattered throughout 2005 to 2018. The inventory included the ID, duration, start and end times for each event.

3.2.3 Rainfall Images Pre-Processing

The pre-processing step for the rainfall images is adopted to extract the relevant timesteps of each storm event from the inventory, impute missing data and remove any anomalous data that would adversely affect the model's performance. This process can be sequentially described by:

(i) **Relevant Timestep Extraction**: The relevant rainfall images pertaining to each storm event in the inventory are extracted from the full radar image dataset. This was achieved by matching the start and end datetimes for each event in the inventory to the relevant timesteps in the full dataset. An example for Event 1 is described and illustrated below.

- Firstly, identify the year in which a given event occurred so that the appropriate datafile corresponding to the same year is utilised. For Event 1, this was the 2014 datafile.
- Secondly, extract all indices from the start to end datetimes for each event from the datafile. For Event 1, the start and end times were (08-Jul-2014 10:15:00) and (08-Jul-2014 20:00:00) respectively. The relevant indices were between 54,268 to 54,385 (i.e. 118 5-minute timesteps).
- Thirdly, the rainfall images corresponding to these indices are subsequently extracted from the datafile. For Event 1, all images from index 54,268 to 54,385 are extracted from the 2014 datafile. Hence, the final dimensions of Event 1 are 110 x 110 x 118, where the first, second and third dimensions are the longitudinal extent, latitudinal extent and timesteps respectively.
- Repeat these steps for all events from the inventory.
- After extraction, there were 39,987 timesteps in total across all 157 events.

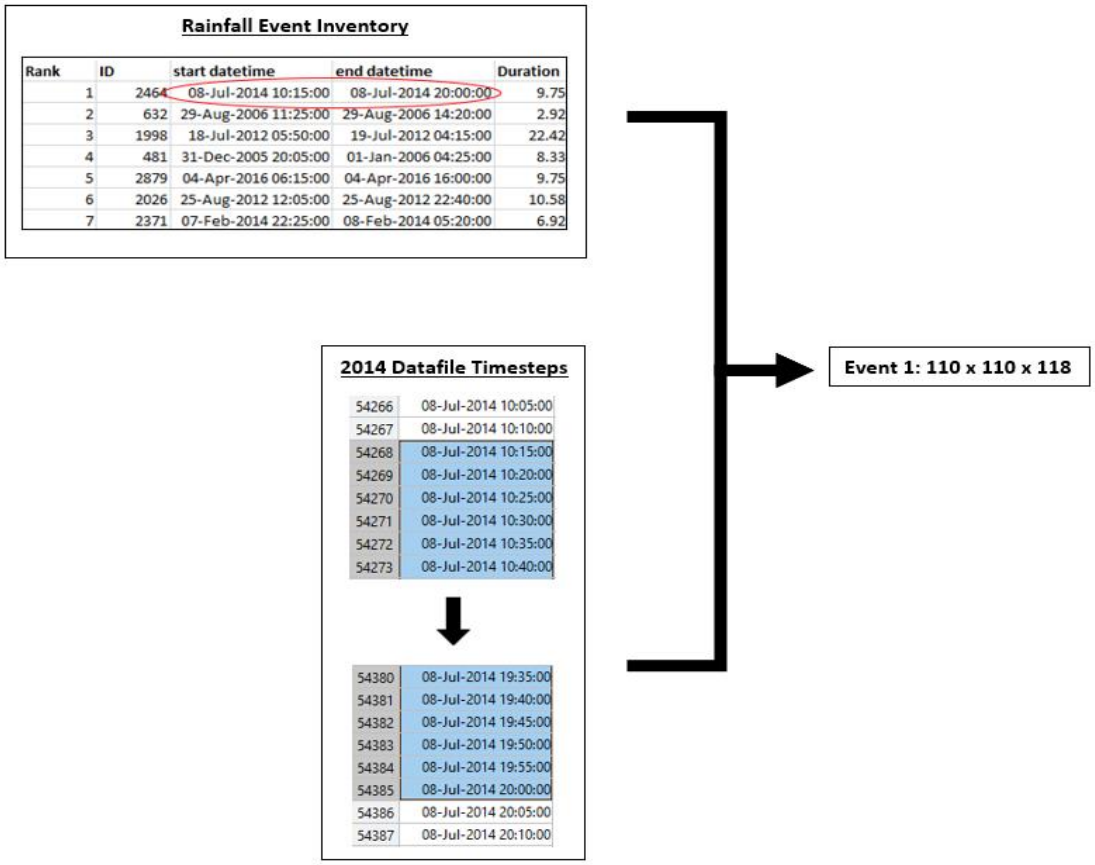


Figure 7 | Extraction of the relevant timesteps from 2014 datafile for Event 1

(ii) **Temporal Imputation**: For each event, if there was any 5-minute interval radar image that was missing, it was imputed by the 5-minute interval radar image from the preceding timestep. The goal here is to preserve as much of the 39,987 timesteps as possible. If too much is discarded, the dataset is too small to train the CNN. Conversely, if too few is discarded, too many missing radar images are imputed thereby adversely affecting the model's performance by indicating the rainfall images are stationary for too long. Hence, if any event had 2 or more missing timesteps, then the event was discarded.

- Missing timesteps were denoted by 110 x 110 matrices of only -1's.
- 24 events were discarded. These corresponded to Events 24, 27, 43, 44, 52, 56, 67, 69, 74, 77, 78, 79, 87, 99, 100, 113, 134, 135, 137, 140, 141, 151, 156, 157
- Of the 700 timesteps in Event 103, there were 66 missing. However, it was the first 66 timesteps that were missing. Therefore, by truncating the event to remove the first 66 timesteps, there were no missing timesteps in the remaining 634. Hence, Event 103 was preserved with 634 timesteps.

(iii) **Spatial Imputation**: For the remaining events, if there was any pixel with a missing value in any of the rainfall images, it was imputed by the average of the surrounding 8 pixels.

- If there was a missing value in any of the surrounding 8 pixels, then that pixel was excluded when computing the average.

- If any rainfall image was missing ≥ 1% of total pixels per image, then that timestep was classified as missing and the number of missing timesteps was increased by 1. This led to 3 additional events being discarded: 64, 66 and 125.
- Hence, 27 events were discarded in total.
- 28,669 rainfall images in total were retained across all 130 events.

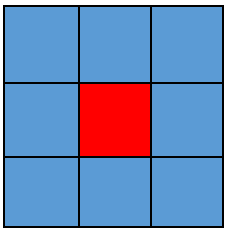


Figure 8 | Missing pixel (red) imputed by the average of the surrounding 8 pixels (blue)

(iv) **Removal of Anomalous Values**: There were pixels that had rainfall intensities exceeding its neighbouring values. Therefore, if any pixel had a rainfall intensity ≥ 250 mm/hr, it was imputed with the average of its neighbouring 8 pixels.
- The maximum rainfall intensity was corrected from over 3000 mm/hr to 249.55 mm/hr.

3.3 Convolutional Neural Networks with Rainfall Images

3.3.1 Database Development

To build this forecasting model, the database on which the CNNs were trained on was comprised of historical rainfall images from all 130 events. Since the goal was to develop a model that would be trained on all events, the rainfall images extracted from each event had to undergo reshaping and feature scaling/transformations such that the rainfall images across events were combined into a single dataset. This was achieved by utilising the nature of 4-dimensional inputs and outputs necessitated by the Keras ML library used to develop CNNs in Python. In each investigation, the inputs and outputs for each model was developed as follows:

(i) **Input and Output Reshaping**: The 1st, 2nd, 3rd and 4th dimensions correspond to the number of samples, image height, image width and number of channels i.e., depth respectively. For all input samples, t_{in} consecutive timesteps were concatenated channel wise i.e., the 4th dimension, where t_{in} is an adjustable parameter denoting the number of timestep inputs. The number of output images was always fixed to 1. For example, if $t_{in} = 3$, then the input images of (t, t+5, t+10) were appended channel wise to forecast the output image at (t+15). This association of input and output forms one sample. Hence, one sample has dimensions: (1, 110, 110, 3) for $t_{in} = 3$. The number of samples per event is given by $(t_{total} - t_{in})$, where t_{total} denotes the total number of timesteps per event. Once $(t_{total} - t_{in})$ samples were created for an event, this process was repeated for all remaining events. The samples across events are then concatenated in the 1st dimension to yield $28669 - 130t_{in}$ samples in total.

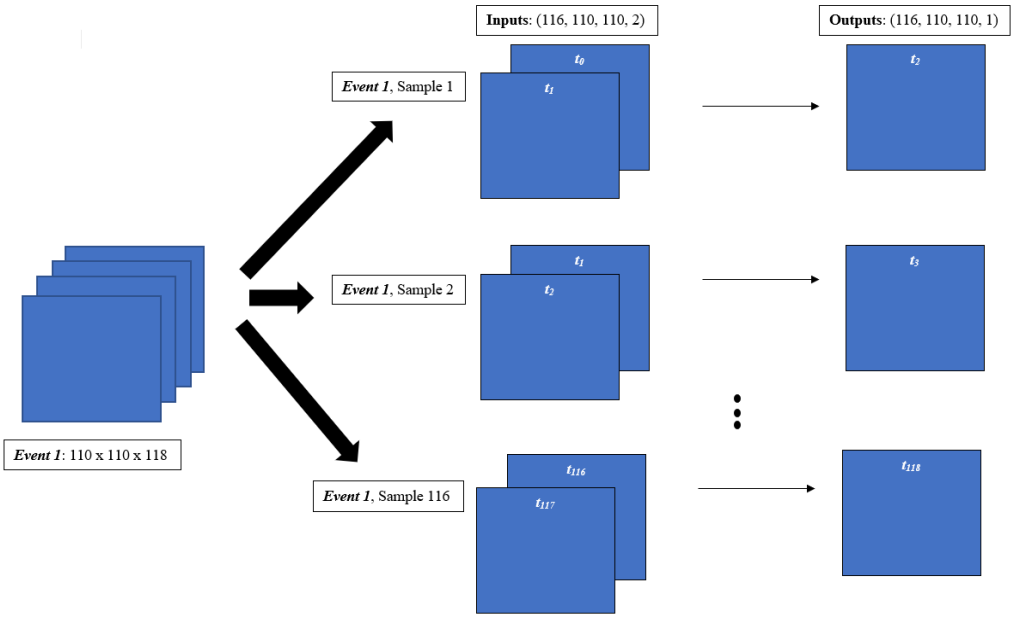


Figure 9 | Illustration of reshaping process for event 1 and $t_{in} = 2$

(ii) **Division into Training, Validation and Test Sets**: According to Brownlee (2017), the training set is used for learning and training to fit the parameters of the model, validation set for fine-tuning model's parameters and test set for evaluating the performance of a fully fine-tuned model. The input (X) and output (Y) datasets were split into 70 – 15 – 15 ratios to the training, validation, and test sets respectively. For example, if t_{in} = 2, the following sizes were observed:

- $X_{training}$: (19886, 110, 110, 2), $Y_{training}$: (19886, 110, 110, 1)
- $X_{validation}$: (4262, 110, 110, 2), $Y_{training}$: (4262, 110, 110, 1)
- X_{test}: (4261, 110, 110, 2), Y_{test}: (4261, 110, 110, 1)

In general, events 1 to 106 were assigned to the training set, events 107 to 126 to the validation set, and events 128 to 155 to the test set.

(iii) **Feature Scaling/Transformations**: Popular methods for feature scaling image pixels include feature normalisation and standardisation. However, Ayzel et. al. (2019) postulates that such methods are only applicable to conventional RGB images for computer vision problems and proposes 2 separate scaling methods for radar-based precipitation nowcasting, which are summarised in Table 1. These transformations were adopted with the prediction that the models would be able to better predict higher rainfall intensities by downscaling them to a smaller value initially. However, it was not evident which transformation method would yield the best results. The predicted outputs are rescaled to the original values by applying the inverse function.

Table 1 | Transformations applied

Transformation	Method	Intensity Range [mm/hr]
Raw	-	[0, 249.55]
1	$ln\,(x_{raw} + 1)$	[0, 5.52]
2	$ln\,(x_{raw} + 0.01)$	[-4.61, 5.52]

3.3.2 Evaluation

Evaluation of a model was conducted by comparing the true outputs from the test set (Y_{test}) with the predicted outputs ($Y_{predicted}$). Since the aim pertains to assessing the prediction of high/extreme rainfall intensities, categorical evaluators derived from a confusion matrix were used.

Table 2 | Confusion Matrix

		True Class [Y]	
		True [Y = 1]	False [Y = 0]
Predicted Class [$\hat{Y}$]	True [$\hat{Y} = 1$]	True Positive (TP)	False Positive (FP)
	False [$\hat{Y} = 0$]	False Negative (FN)	True Negative (TN)

The rainfall intensity of any pixel was classified as 'high' i.e., Y = 1, if the true or predicted intensity is $\geq 99.9^{th}$ percentile of all non-zero intensities in the 34km x 30km area of interest across all events. This threshold intensity corresponded to 40.84 mm/hr and according to the UK Met Office (2011), rainfall intensities within 10 – 50 mm/hr correspond to 'heavy' rainfall. Therefore, a threshold of 40.84 mm/hr is appropriate as a 'high' rainfall indicator since it can be considered very heavy rainfall.

Table 3 | Outcomes

Classification	**Condition**	**Description**
True Positive (TP)	If $y_i^{true} \geq 40.84$ mm/hr and $y_i^{predicted} \geq 40.84$ mm/hr	Extreme rainfall correctly predicted
False Negative (FN)	If $y_i^{true} \geq 40.84$ mm/hr and $y_i^{predicted} < 40.84$ mm/hr	Extreme rainfall predicted as non-extreme rainfall
False Positive (FP)	If $y_i^{true} < 40.84$ mm/hr and $y_i^{predicted} \geq 40.84$ mm/hr	Non-extreme rainfall predicted as extreme rainfall
True Negative (TN)	If $y_i^{true} < 40.84$ mm/hr and $y_i^{predicted} < 40.84$ mm/hr	Non-extreme rainfall correctly predicted

Note: y_i^{true} and $y_i^{predicted}$ denote the true and predicted rainfall intensity of the i-th pixel in the test set respectively.

The number of TP, FN, FP, and TN are tallied from the test set and 3 quantitative metrics in addition to qualitative analysis are calculated for evaluation. These are summarised in Table 4. The chosen metrics are associated with the positive hit rates i.e., the hit rates on extreme rainfall pixels since it aligns with this project's aims. Qualitative evaluations are also adopted to determine if predicted outputs resembled the true spatial structures and positions, which are difficult to assess quantitatively. For example, if the model predicted a high rainfall pixel adjacent to its true position, then this prediction would be flagged as incorrect. However, this is within reasonable limits and the model should therefore not be penalised. Although more subjective, the combination of both quantitative and qualitative analysis provides a more holistic depiction of the model's performance.

Table 4 | Quantitative metrics

Metrics	Equation	Description
Recall (R)	$\frac{TP}{TP + FN}$	Hit rate of correctly predicting extreme rainfall pixels to all true extreme rainfall pixels
Precision (P)	$\frac{TP}{TP + FP}$	Ratio of correctly predicted extreme rainfall pixels to all predicted extreme rainfall pixels
F1-Score (F)	$2\frac{P \times R}{P + R}$	Harmonic mean of Precision (P) and Recall (R)

These metrics were chosen due to the skewness of the data. Namely, zero intensity pixels significantly outnumbers non-zero intensity pixels, which is verified by Table 5. By using metrics such as the accuracy (the overall correct prediction rate for extreme and non-extreme rainfall pixels), there would be an inaccurate representation of the performance. For example, if 99% of pixels are non-extreme and 1% of pixels are extreme, a model that exclusively forecasts zero intensity pixels would have an accuracy of 99%. Therefore, P and R are used instead. However, it is difficult to gauge which model configurations have performed better using 2 separate metrics i.e., some models may have higher P but lower R. Hence, it is common to combine P and R into F by taking the harmonic mean. A perfect model will have F = 1 and in general, the higher F is, the better as this indicates high P and R.

Table 5 | Pixel intensity distribution

Intensity Range [mm/hr]	Number of Pixels	% of Total Pixels
$x = 0$	190,447,270	54.90%
$0 < x \leq 0.1$	25,565,841	7.37%
$0.1 < x \leq 1$	79,414,081	22.89%
$1 < x \leq 10$	50,177,108	14.46%
$10 < x \leq 100$	1,284,599	0.37%
$x > 100$	6001	Negligible

3.3.3 Convolutional Neural Network 1

Convolutional Neural Network 1 (CNN_1) was one of two models investigated that are exclusively trained on rainfall images only. The general architecture for CNN_1 was inspired by the model from Ayzel et. al. (2019) in their investigation of using CNNs for radar-based precipitation nowcasting. A summary of the model architecture is shown in Table 6 and illustrated in Figure 10. The chosen optimizer was Adam (Adaptive Moment Estimation) and the number of epochs was varied in each investigation depending on the computational efforts and times. A baseline model of CNN_1 was developed with the filter size = 3 and t_{in} = 2 such that performance comparisons could be made between this baseline model and its counterparts.

Table 6 | Summary of layers for CNN_1

Layer	Layer Type	Padding	Filter Size	Stride	Activation Function	$N_{FILTERS}$
CONV1	Convolutional	Same	3	1	ReLU	48
CONV2						24
CONV3						12
CONV4						6
CONV5						3
CONV6					-	1

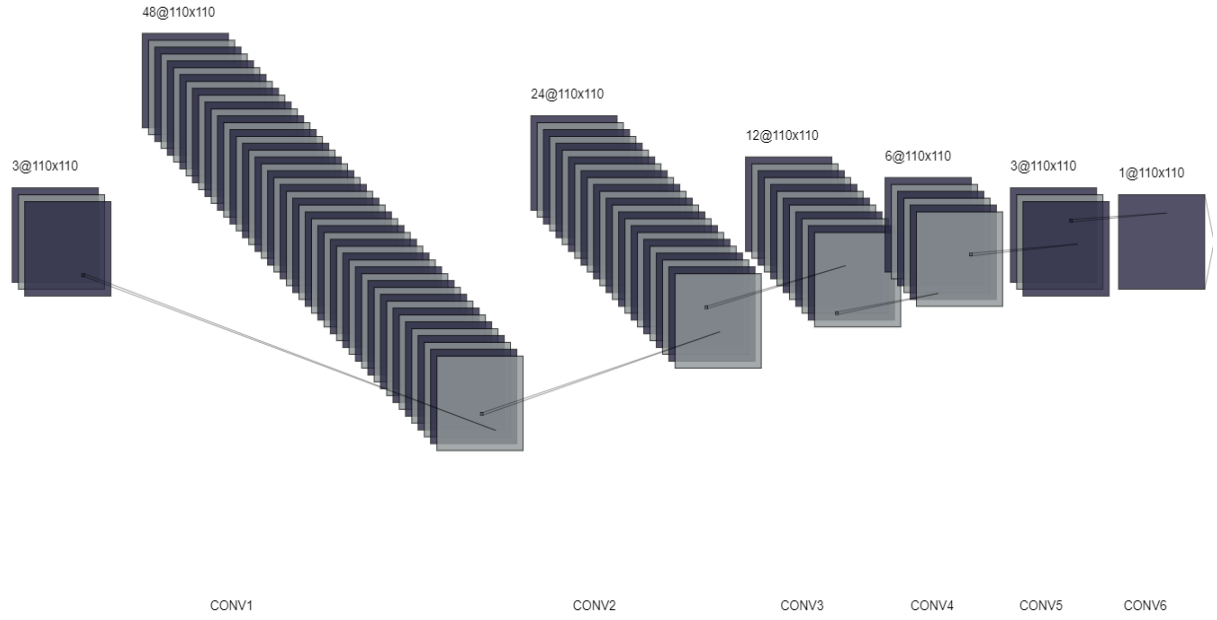


Figure 10 | Model architecture of CNN_1 (t_{in} = 3)

3.3.4 Hyperparameter Tuning and Investigations

3.3.4.1 Transformations and Loss Functions

Since forecasting pertains to a regression problem, Grover (2018) states that popular loss functions include the Mean Absolute Error (MAE), Mean Squared Error (MSE) and Logcosh. However, MAE is more robust to outliers than MSE and Ayzel et. al. (2019) also obtained worse results in their investigation of radar-based precipitation nowcasting with MSE than MAE. Although it would be more appropriate to train the model on a loss function that measures the deviation between the true and predicted intensities for high rainfall pixels only, the Keras ML library has no specific loss function pertaining to this and thus presented a software issue. Therefore, only the MAE and Logcosh loss functions were investigated.

$$MAE = \frac{\sum_{i=1}^{n} |y_i^{true} - y_i^{predicted}|}{n} \quad (2)$$

$$LOGCOSH = \sum_{i=1}^{n} ln\left(cosh\left(y_i^{true} - y_i^{predicted}\right)\right) \quad (3)$$

However, it was not evident which loss function would perform the best in each investigation. When coupled with the uncertainty as to which combination of loss function and transformation method would work best, the number of permutations becomes substantial, implicating long computational times. Therefore, to minimise the number of permutations to investigate, the best performing combination of loss function and transformation method was first identified for CNN_1 and all subsequent investigations of CNN_1 was configured to this combination. The assumption here is that the best performing combination would remain true for all investigations.

Table 7 | Combinations of loss function and transformation for CNN_1

Combination	Loss Function	Transformation
1	MAE	Raw
2		Transformation 1
3		Transformation 2
4	Logcosh	Raw
5		Transformation 1
6		Transformation 2

3.3.4.2 Filter Sizes

In the baseline model, the filter size hyperparameter was fixed to 3. It was not apparent how changing the filter size impacts the model's performance. Therefore, this was examined by increasing the filter size to 5. Comparison are also made to the baseline model to quantify any impacts on the model's performance.

3.3.4.3 Spatial Aggregations

In some instances, the data used in addition to rainfall images for flood predictions are of coarser resolutions and to incorporate these inputs together into a model, a homogeneous spatial resolution is required. Typically, this involves scaling up the finer resolutions to match the coarser resolutions. Hence, the effect of spatially aggregating pixels was investigated by taking 5x5 spatial averages, which reduced the spatial area from 110x110 to 22x22. The impacts of increasing to a 5km x 5km resolution was quantified and compared to the baseline model (1km x 1km resolution). The 34x30 area of interest was also reduced to a 7x6 area.

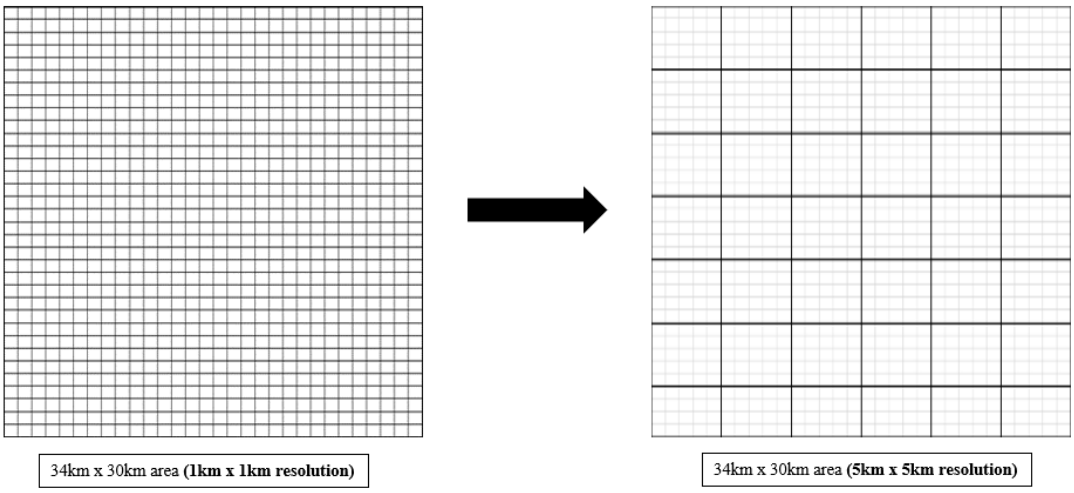


Figure 11 | 5x5 aggregation of pixel squares

3.3.4.4 Reduced Image Size

It was theorised that applying 'same' padding to the 34x30 area would be 'misleading' as it implies no information is available outside this area. Therefore, it was hypothesised that providing information beyond the 34x30 area using 110x110 images instead would not 'mislead' the model, which leads to a better performance. To verify this, a comparison was made between training the model on 110x110 images and 34x30 images.

3.3.4.5 Varying t_{in}

In the baseline model, t_{in} was set to 2. However, it was hypothesised that increasing t_{in} would improve the model's overall performance because by providing more images, more information would be inputted per sample. Hence, t_{in} was increased to 3 and 4. Table 8 summarises the input setups.

Table 8 | Summary of different t_{in} setups

t_{in}	Input Images	Output Image
2	[t, t+5]	t+10
3	[t, t+5, t+10]	t+15
4	[t, t+5, t+10, t+15]	t+20

3.3.4.6 Timestep Irregularity

In practice, it is not uncommon to have missing rainfall images from the dataset, which cannot be imputed with preceding timesteps. An example of this is severe weather systems that can travel significant distances and change shape rapidly every 5-minutes. Thus, imputing with preceding timesteps implies stationarity, which is misleading and leads to erroneous predictions. Although the same temporal imputation method from pre-processing could be used, there is an abundance of information from severe weather events that can be learnt and thus discarding events with > 1 missing timestep is impractical. Hence, the model's robustness under these circumstances needs to be assessed. In effect, timestep irregularity can be simulated by mixing the temporal resolution of the input sequence. Therefore, 3 different input permutations for t_{in} = 2 and 3 were investigated, which are summarised in Table 9.

Table 9 | Permutations of timestep irregularity for t_{in} = 2 and 3

Permutation	t_{in}	Input Images (X)	Output Image (Y)
1 [P_1]	2	[t, t+10]	t+15
2 [P_2]	3	[t, t+5, t+15]	t+20
3 [P_3]		[t, t+10, t+15]	t+20

3.3.4.7 Increased Forecast Lead Time

When designing a flood warning system that relies on the outputs from rainfall forecasts as inputs, a lead time of 5-minutes is too narrow for any warning disseminations. In practice, flood warning lead times are typically hours or days, and rarely minutes. Hence, as an investigation, the lead time was increased to 10 and 15-minutes and the model's performance was compared to a lead time of 5-minutes. Table 10 summarises the setups.

Table 10 | Investigations for increased forecast lead times

Input Images	Forecast Lead Time	Output Image
[t, t+5]	5-minutes	t+10
	10-minutes	t+15
	15-minutes	t+20

3.3.4.8 Increased Temporal Resolution

In some situations, the temporal resolution between rainfall images may not be 5-minutes. The reasons for this are the sampling costs at such frequencies, limited data storage, etc. Hence, it is imperative to quantify any impacts on the model's performance with an increased temporal resolution. A comparison was made between the baseline model (5-minutes) and a model trained on 15-minute rainfall images, which was achieved by taking every 3rd image from each event. t_{in} was set to 2 in both cases.

3.3.5 Convolutional Neural Network 2

Convolutional Neural Network 2 (CNN_2) is the second model that is trained on rainfall images only. The architecture of CNN_2 was inspired by Kosson, Mu and Wang (2017) in their investigation of using CNNs for action conditioned video game frame predictions. This model is also used for the investigation on incorporating climatological data with rainfall images (i.e., CNN_3). To ensure a fair comparison, it was necessary to investigate if the same model could perform without the climate data by eliminating the climate branch of the network and hence, be trained on rainfall images only. Table 11 summarises the layers of CNN_2 and Figure 12 provides an illustration.

Table 11 | Summary of CNN_2 for 110 x 110 x t_{in} image inputs

Layer	Layer Type	$N_{NEURONS}$	Padding	FS	Stride	AF	$N_{FILTERS}$
CONV1	Convolutional	-	Valid	8	2	ReLU	64
CONV2				6			128
CONV3				6			128
CONV4				6			128
FC1	Fully Connected	1024	-	-	-	ReLU	-
FC2		2048				Linear	
FC3		1024				Linear	
FC4		1152				ReLU	
DCO1	Deconvolutional	-	Valid	6	2	ReLU	128
DCO2				6			128
DCO3				6			64
DCO4				8		-	1

NOTE: Deconvolutional layers are merely the inverse/reverse of convolutional layer. FS and AF denote filter size and activation function respectively.

To 'feed' the convolved feature from CONV4 into FC1, it undergoes flattening. If the convolved feature from CONV4 is of dimensions (Height [H], Width [W], Channels [C]), it is flattened into a ($H \times W \times C$, 1) input vector, which is then fed into FC1. Likewise, the output from FC4 is reshaped into (H, W, C) prior to feeding it into DCO1. This flattening and reshaping process is denoted by the black arrows in Figure 12.

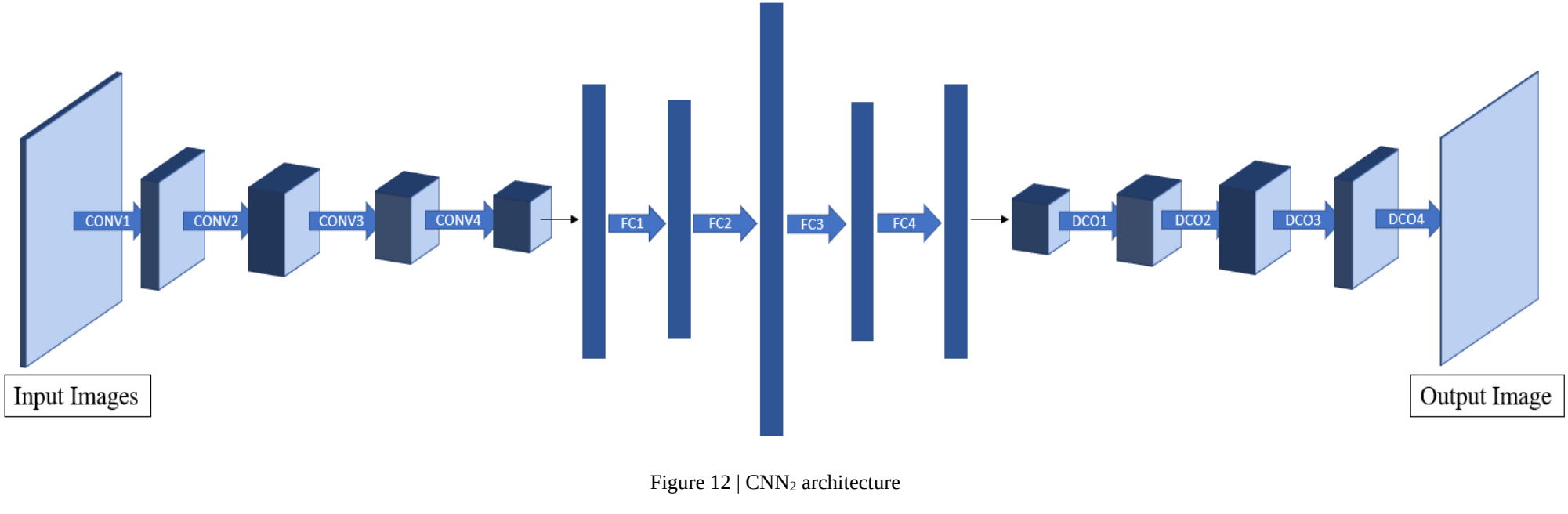


Figure 12 | CNN_2 architecture

3.4 Convolutional Neural Networks with Rainfall Images and Climate Data

3.4.1 Convolutional Neural Network 3

Convolutional Neural Network 3 (CNN_3) is the network that combines climatological data and rainfall images as inputs to forecast subsequent rainfall images. CNN_3 was inspired by the model architecture proposed by Kosson, Mu and Wang (2017). In their investigations, frames of a video game were combined with a player's actions in each frame through a separate branch of the network to predict subsequent frames. For this project, the model was adapted by substituting video game frames for rainfall images and player's actions for climatological data. Tables 12, 13 and 14 and Figure 13 illustrate CNN_3.

Table 12 | Section of CNN_3 that processes/convolves the rainfall images

Layer	Layer Type	$N_{NEURONS}$	Padding	FS	Stride	AF	$N_{FILTERS}$
CONV1	Convolutional	-	Valid	8	2	ReLU	64
CONV2				6			128
CONV3				6			128
CONV4				6			128
FC1	Fully Connected	1024	-	-	-	ReLU	-
FC2		2048				Linear	

Table 13 | Section of CNN_3 for climate data (Climate branch)

Layer	Layer Type	$N_{NEURONS}$	AF
FC3	Fully Connected	2048	Linear

(2048, 1) vector from FC2

(2048, 1) vector from FC3

Element Wise Multiplication

Table 14 | Section of CNN_3 that processes the combined climate and rainfall data to predict the next rainfall image

Layer	Layer Type	$N_{NEURONS}$	Padding	FS	Stride	AF	$N_{FILTERS}$
FC4	Fully Connected	1024	-	-	-	Linear	-
FC5		1152				ReLU	-
DCO1	Deconvolutional	-	Valid	6	2	ReLU	128
DCO2				6			128
DCO3				6			64
DCO4				8		-	1

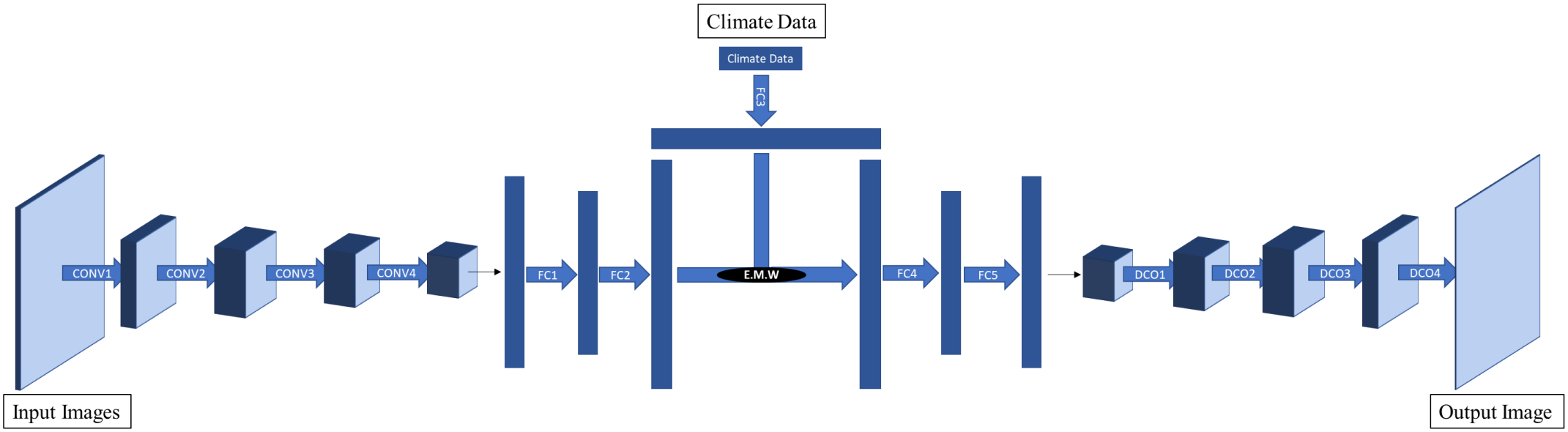


Figure 13 | Model architecture for CNN_3 (E.M.W denotes Element Wise Multiplication)

3.4.2 Climate Data

The chosen climate dataset was the ERA5 hourly data at a pressure level of 500hPa due to its strong relationship with rainfall features (Chen, 2021). The datafile was obtained from the European Centre for Medium-Range Weather Forecasts (ECMWF) and spatially encompasses latitudes and longitudes from (54.00, -3.00) to (51.00, 0.00) with a spatial resolution of (-0.25°, +0.25°). The data also covers the time span from 1st January 1979 00:00:00 to 31st December 2018 23:00:00 with a temporal resolution of 1 hour. The full climate dataset contains 7 variables:

Table 15 | Summary of climatological variables in the climate dataset

Variable	Notation	Units [-]
Geopotential height	z	m^2s^{-2}
Ozone mixing ratio	o_3	-
Relative humidity	r	%
Specific rainwater content	crwc	-
Temperature	t	K
U-component of wind (eastward)	u	ms^{-1}
V-component of wind (northward)	v	ms^{-1}

3.4.3 Rainfall Images Pre-Processing

Some events had start and end times that were not on the hour. Therefore, after the rainfall data was pre-processed as described in 3.2.3, each event was 'truncated' such that the new start and end times of each event was on the hour. Subsequently, only the rainfall images between the new hourly start and end times are kept. This truncation step is conducted for reasons explained after Table 16.

Furthermore, the mismatch between temporal resolutions of the rainfall and climate data also introduces a new parameter in addition to t_{in} – the window size, which dictates how many timesteps to shift along after each sample. For example, with 10-minute windows and $t_{in} = 2$, sample 1 uses images at minutes 0 and 5 to predict the image at minute 10 but sample 2 uses images at minutes 10 and 15 to predict the image at minute 20. In the case of CNN_1, the window size was always 5-minutes whereas with CNN_3, this parameter is not fixed. Nevertheless, Table 16 summarises all possibilities of how the original start and end times of each event are truncated.

Table 16 | Truncation possibilities for all events

If start time is…	If end time is…	Truncation method
On the hour	Not on the hour	Maintain start time, round down end time to nearest hour (e.g., 11:45 → 11:00)
Not on the hour	On the hour	Round up start time to nearest hour (e.g., 10:15 → 11:00), maintain end time
Not on the hour	Not on the hour	Round up start time to nearest hour, round down end time to nearest hour
On the hour	On the hour	Maintain both start and end times

The start times are rounded up because images preceding the original start time are not associated with the event. A similar intuition applies to the images after the original end time. This resulted in 3 additional events being discarded: 70, 81 and 126 since these events had equal start and end times after truncation. Hence, 127 events remained, and the total number of rainfall images was reduced from 28,669 to 27,066. This truncation step to achieve hourly start and end times was conducted to facilitate training of CNN_3's baseline model on 1-hour windows with $t_{in} = 12$. In short, this means images corresponding to minutes 0 to 55 are always used as inputs to forecast the image of the next hour for all samples (e.g., images 10:00:00 to 10:55:00 are combined with the climate data at 10:00:00 to forecast the 11:00:00 rainfall image, images 11:00:00 to 11:55:00 are combined with the climate data at 11:00:00 to forecast the 12:00:00 rainfall image, etc.).

1-hour windows with $t_{in} = 12$ was chosen for the baseline model of CNN_3 for two reasons. Firstly, it avoids any overlap between consecutive climate data points, which complicates the input reshaping process. To illustrate this 'overlap' issue, consider a 30-minute window with $t_{in} = 12$. When the input images are 10:00:00 to 10:55:00, then only the climate data at 10:00:00 is relevant to these 12 input images. In the next sample, the input images are 10:30:00 to 11:25:00, where there is now an overlap between the climate data at 10:00:00 and 11:00:00. This is because images from 10:30:00 to 10:55:00 are associated with the 10:00:00 climate data and images from 11:00:00 to 11:25:00 are associated with the 11:00:00 climate data. This example demonstrates how the shape of the inputted climate data will repeatedly vary across samples (i.e., sometimes 2 consecutive climate data points are relevant but sometimes only 1), which complicates the input reshaping process since the Keras library does not accept varying input sizes across samples. Therefore, by avoiding this overlap using 1-hour windows with $t_{in} = 12$, the input reshaping process is simplified since all samples only have 1 climate data point associated with its 12 input images. Nevertheless, this issue is not disregarded and is further examined as an investigation of CNN_3 in 3.4.7.3.

Secondly, 1-hour windows with $t_{in} = 12$ also reduces computational times, albeit at the expense of reducing the number of training samples. This is important in offsetting the increase in computational times due to the increase in the number of learnable parameters, which was now 9,539,457 with CNN_3 in comparison to 14,755 for CNN_1. Additionally, there are three times as many combinations of loss functions and transformations to be examined on the baseline model of CNN_3 than CNN_1, as described in 3.4.7.1. Thus, adopting a 1-hour window with $t_{in} = 12$ for the baseline model will also help accelerate the investigations of 3.4.7.1 in addition to simplifying the input reshaping process.

3.4.4 Climate Data Pre-Processing

The pre-processing step for the climate data was conducted to extract the relevant spatial area and timespans. The process can be sequentially described as follows:

(i) **Extraction of Relevant Spatial Area from Climate Data**: Since the spatial extent of the climate data was greater than the spatial extent of the rainfall data, the spatial extent of the rainfall data i.e., (350,000, 230,000) to (459,000, 339,000) was first converted into latitude and longitude pairs so that the relevant climate data points corresponding to the spatial extent of the rainfall data could be identified and extracted. This resulted in 28 latitude and longitude pairs.

Table 17 | Latitudes and longitudes encompassed by the spatial extent of the rainfall images

Relevant Latitudes [°]	Relevant Longitudes [°]
52.75, 52.5, 52.25, 52.00	-2.75, -2.50, -2.25, -2.00, -1.75, -1.5, -1.25

(NOTE: To get 28 pairs, each 'relevant latitude' must be paired with every 'relevant longitude')

			z	o3	r	crwc	t	u	v
latitude	longitude	time							
52.75	-2.75	1979-01-01 00:00:00	50939.070312	8.124012e-08	10.831995	0.000000e+00	238.234406	8.480528	-7.839794
		1979-01-01 01:00:00	50994.589844	7.969649e-08	11.436916	0.000000e+00	237.459869	7.192266	-8.995173
		1979-01-01 02:00:00	51017.210938	7.753621e-08	11.774729	0.000000e+00	236.584915	5.309841	-10.688672
		1979-01-01 03:00:00	51074.367188	7.546743e-08	12.032017	0.000000e+00	235.584152	3.016770	-12.605730
		1979-01-01 04:00:00	51116.191406	7.338671e-08	12.904046	0.000000e+00	234.396667	0.349395	-14.362538
...	...	...	...	...	...	...	...	...	...
52.00	-1.25	2018-12-31 19:00:00	57036.640625	8.819542e-08	23.557253	-5.293956e-23	255.785461	14.179055	-0.691568
		2018-12-31 20:00:00	57017.648438	8.971404e-08	38.352398	-5.293956e-23	255.450500	15.055783	-2.185944
		2018-12-31 21:00:00	56965.773438	9.054444e-08	45.161343	-5.293956e-23	255.191879	16.128487	-2.986862
		2018-12-31 22:00:00	56934.023438	9.167061e-08	52.687386	-5.293956e-23	254.887192	18.040098	-3.403272
		2018-12-31 23:00:00	56906.667969	9.157392e-08	47.597614	-5.293956e-23	254.904861	18.222321	-4.637392

9817920 rows × 7 columns

Figure 14 | Relevant spatial areas extracted from climate dataset

(ii) **Extraction of Relevant Timespans from Climate Data**: All 127 storm events were scattered throughout 2005 to 2018 so the period between 1st January 2005 00:00:00 to 31st December 2018 23:00:00 was extracted from the climate data.

latitude	longitude	time	z	o3	r	crwc	t	u	v
52.75	-2.75	2005-01-01 00:00:00	55372.246094	6.180700e-08	9.975124	0.000000e+00	253.969177	26.709398	3.616792
		2005-01-01 01:00:00	55314.753906	6.167026e-08	12.046572	0.000000e+00	253.842285	25.262150	1.168194
		2005-01-01 02:00:00	55260.875000	5.842269e-08	25.201960	0.000000e+00	253.817017	24.392366	0.748279
		2005-01-01 03:00:00	55199.335938	5.990241e-08	33.413040	0.000000e+00	253.560349	25.107920	1.793523
		2005-01-01 04:00:00	55131.589844	6.181188e-08	37.945324	0.000000e+00	253.037827	25.197588	2.951471
...	...	...	...	...	...	...	...	...	...
52.00	-1.25	2018-12-31 19:00:00	57036.640625	8.819542e-08	23.557253	-5.293956e-23	255.785461	14.179055	-0.691568
		2018-12-31 20:00:00	57017.648438	8.971404e-08	38.352398	-5.293956e-23	255.450500	15.055783	-2.185944
		2018-12-31 21:00:00	56965.773438	9.054444e-08	45.161343	-5.293956e-23	255.191879	16.128487	-2.986862
		2018-12-31 22:00:00	56934.023438	9.167061e-08	52.687386	-5.293956e-23	254.887192	18.040098	-3.403272
		2018-12-31 23:00:00	56906.667969	9.157392e-08	47.597614	-5.293956e-23	254.904861	18.222321	-4.637392

3435936 rows × 7 columns

Figure 15 | Relevant timesteps extracted from climate dataset

(iii) **Imputation of Missing Data**: Any missing data point in each climate variable was denoted by -32767. However, it was found that there was no missing data in any variable and thus no imputation was required.

(iv) **Spatial Averaging**: To eliminate the spatial dimension of the climate dataset, a spatial average across all 28 latitude and longitude pairs at each hourly timestep for all timesteps.

(v) **Elimination of Redundant Variables**: This pertains to any variable with only 1 outcome. These variables are discarded because there is no useful information on how the dependent variable varies with this independent variable, which adversely affects the model's performance. In the case of the climate data, this was crwc as there were only 2 results: either zero or -5.29×10^{-23} and since 10^{-23} is sufficiently small to be approximated to zero, it could be assumed that crwc only had 1 result. Thus, crwc was removed from the climate dataset and 6 variables remained: z, o_3, r, t, u, and v.

time	z	o3	r	t	u	v
2005-01-01 00:00:00	55372.246094	6.180700e-08	9.975124	253.969177	26.709398	3.616792
2005-01-01 01:00:00	55314.753906	6.167026e-08	12.046572	253.842285	25.262150	1.168194
2005-01-01 02:00:00	55260.875000	5.842269e-08	25.201960	253.817017	24.392366	0.748279
2005-01-01 03:00:00	55199.335938	5.990241e-08	33.413040	253.560349	25.107920	1.793523
2005-01-01 04:00:00	55131.589844	6.181188e-08	37.945324	253.037827	25.197588	2.951471
...	...	...	...	...	...	...
2018-12-31 19:00:00	57036.640625	8.819542e-08	23.557253	255.785461	14.179055	-0.691568
2018-12-31 20:00:00	57017.648438	8.971404e-08	38.352398	255.450500	15.055783	-2.185944
2018-12-31 21:00:00	56965.773438	9.054444e-08	45.161343	255.191879	16.128487	-2.986862
2018-12-31 22:00:00	56934.023438	9.167061e-08	52.687386	254.887192	18.040098	-3.403272
2018-12-31 23:00:00	56906.667969	9.157392e-08	47.597614	254.904861	18.222321	-4.637392

Figure 16 | Spatially averaged climate dataset with crwc removed.

3.4.5 Database Development

The rainfall images and climatological data had to undergo further reshaping to create the necessary inputs and outputs for CNN_3.

(i) **Rainfall Image Input and Output Reshaping**: In the case of the rainfall images, the input and output images were reshaped following the method described in 3.3.1 but with t_{in} = 12 and 1-hour windows for the baseline model. Therefore, per sample, the input images (X) and output image (Y) had dimensions: (1, 110, 110, 12) and (1, 110, 110, 1) respectively. Hence, with t_{in} = 12 and 1-hour windows, the full dimensions of the inputs and outputs across all 127 events were: (2258, 110, 110, 12) and (2258, 110, 110, 1) respectively.

(ii) **Event Matching for Climate Data**: The hourly timesteps pertaining to each storm event are extracted from the spatially averaged climate dataset. This was achieved by matching the truncated start and end times for each event to the relevant hourly timesteps. An example for Event 1 is described and illustrated below.

- Event 1 original start and end times: 8th July 2014 10:15:00 to 8th July 2014 20:00:00
- Truncating Event 1 means the start and end times are shifted to: 8th July 2014 11:00:00 to 8th July 2014 20:00:00 (End time did not shift because it's already on the hour).
- Hence, the relevant input timesteps from the climate dataset on 8th July 2014 are: 11:00:00, 12:00:00, 13:00:00, 14:00:00, 15:00:00, 16:00:00, 17:00:00, 18:00:00, 19:00:00.
- These timesteps and all 6 climatological features were extracted and appended row wise into a table as shown in Figure 17.
- 20:00:00 is not included because it is the last timestep of the event and hence, the last output – the last sample of event 1 uses input images 19:00:00 to 19:55:00 to predict 20:00:00 and thus, only the climate data for 19:00:00 is needed.

time	z	o3	r	t	u	v
2014-07-08 11:00:00	54797.8	1.05872e-07	80.1109	250.661	1.44206	4.43165
2014-07-08 12:00:00	54805.5	1.03721e-07	87.4512	250.486	0.702792	3.90202
2014-07-08 13:00:00	54817.7	1.00963e-07	92.7424	250.412	-0.259583	3.49454
2014-07-08 14:00:00	54839.7	9.86153e-08	97.9168	250.352	-1.32529	2.64793
2014-07-08 15:00:00	54874.6	9.88645e-08	96.0909	250.496	-2.83949	1.92017
2014-07-08 16:00:00	54909.2	1.0448e-07	84.2095	250.941	-4.11528	1.77241
2014-07-08 17:00:00	54975.8	1.09652e-07	73.283	251.513	-4.2584	1.74741
2014-07-08 18:00:00	55019.7	1.11433e-07	72.5184	251.785	-3.94098	1.07806
2014-07-08 19:00:00	55073.8	1.12507e-07	72.4546	251.995	-3.54425	-0.301483

Figure 17 | Event 1 climate data

- This process was subsequently repeated for all events and the extracted timesteps are appended row wise across events to obtain an event indexed climate dataset.

		z	o3	r	t	u	v
event number	time						
Event 1	2014-07-08 11:00:00	54797.8	1.05872e-07	80.1109	250.661	1.44206	4.43165
	2014-07-08 12:00:00	54805.5	1.03721e-07	87.4512	250.486	0.702792	3.90202
	2014-07-08 13:00:00	54817.7	1.00963e-07	92.7424	250.412	-0.259583	3.49454
	2014-07-08 14:00:00	54839.7	9.86153e-08	97.9168	250.352	-1.32529	2.64793
	2014-07-08 15:00:00	54874.6	9.88645e-08	96.0909	250.496	-2.83949	1.92017
...	...	...	...	...	...	...	...
Event 155	2005-12-30 12:00:00	52873.3	6.71936e-08	100.677	252.373	24.1371	4.18837
	2005-12-30 13:00:00	52745.3	6.58832e-08	100.453	251.808	24.9069	7.70662
	2005-12-30 14:00:00	52644.9	7.00447e-08	92.4219	250.941	26.4277	8.94974
	2005-12-30 15:00:00	52536.6	7.86258e-08	77.1889	249.87	27.5632	7.93105
	2005-12-30 16:00:00	52422	8.00961e-08	69.9449	249.191	24.6988	8.19029

Figure 18 | Event indexed climate dataset with 6 climatological features

- The final dimension of the event indexed climate dataset was (2258, 6) for 1-hour windows and t_{in} = 12. The 1st dimension of the event indexed climate dataset should equal the total number of input samples (X). Doing so ensures that the climatological data pertaining to the same timestep as the rainfall images are associated together across all events.

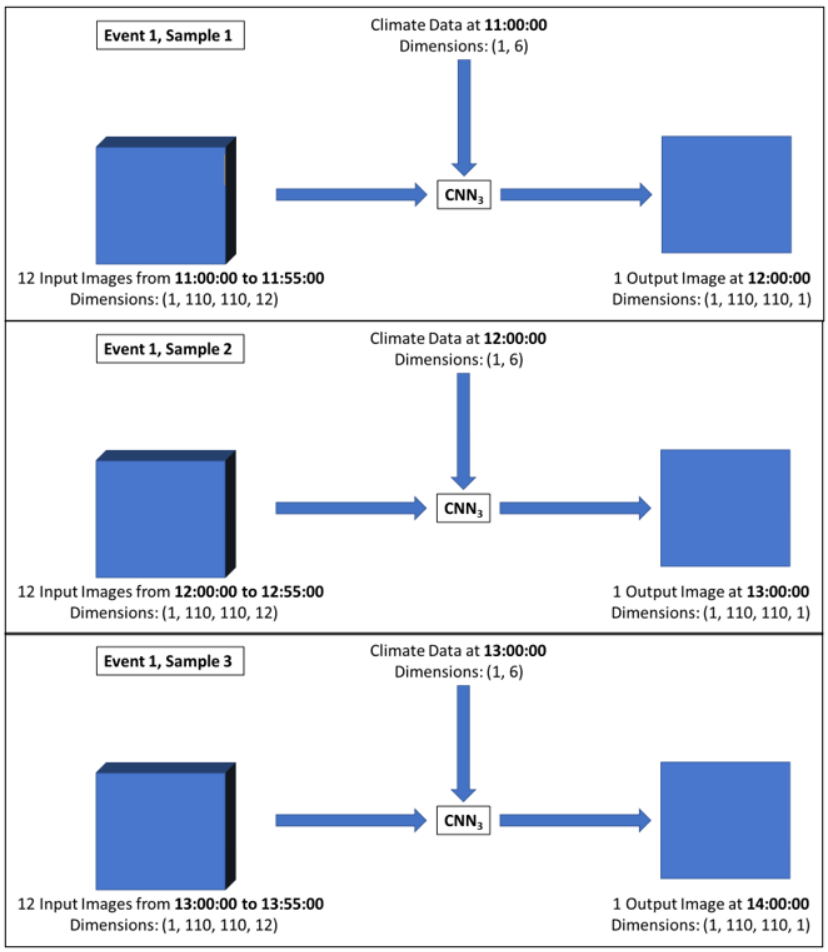


Figure 19 | Illustration of how the climate data and rainfall images are utilised in the baseline model of CNN_3

3.4.6 Evaluation

Precision (P), Recall (R) and the F1-Score (F) along with qualitative analysis are utilised again for evaluating CNN_3.

3.4.7 Hyperparameter Tuning and Investigations

3.4.7.1 Transformations and Loss Functions

As with CNN_1, it was not evident which loss function and rainfall intensity transformation pair would perform the best for CNN_3. With the addition of climate data, three different climatological transformations were also investigated: raw, feature normalisation (min-max scaling) and equalisation (rescaling each variable to the same range as the rainfall intensities). Hence, there are 18 combinations to investigate in total, which therefore made it imperative to reduce computational times in order to accelerate this investigation. This was achieved by using 1-hour windows and $t_{in} = 12$ as described in 3.4.3. Once the best performing combination was identified from the baseline model, all subsequent investigations of CNN_3 would be configured to this combination.

Table 18 | Summary of the 18 combinations investigated

Combination	Loss Function	Rainfall Transformation [Range]	Climate Transformation [Range]
1	MAE	Raw [0, 249.55]	Raw
2			Normalised [0, 1]
3			Equalised [0, 249.55]
4		Transformation 1 [0, 5.52]	Raw
5			Normalised [0, 1]
6			Equalised [0, 5.52]
7		Transformation 2 [-4.61, 5.52]	Raw
8			Normalised [0, 1]
9			Equalised [-4.61, 5.52]
10	Logcosh	Raw [0, 249.55]	Raw
11			Normalised [0, 1]
12			Equalised [0, 249.55]
13		Transformation 1 [0, 5.52]	Raw
14			Normalised [0, 1]
15			Equalised [0, 5.52]
16		Transformation 2 [-4.61, 5.52]	Raw
17			Normalised [0, 1]
18			Equalised [-4.61, 5.52]

3.4.7.2 Varying Input Setup (t_{in} and Window Size)

Once the best performing combination from 3.4.7.1 was identified, the number of training samples was then increased by varying the input setup to appropriately quantify the model's performance. However, the climate data was limiting since its temporal resolution was 1-hour in comparison to the 5-minutes of rainfall images. Hence, to 'artificially' increase the sample size, 2 setups were investigated – one with 'overlap' between consecutive hourly climate data points and one without. For no overlap, this was achieved by using $t_{in} = 3$ and a window size = 5-minutes whilst duplicating the climate data 10 times for every hour across all events and discarding every 11^{th} and 12^{th} sample. An example is shown in Table 19.

Table 19 | Rainfall image samples from 11:00:00 of an arbitrary event

Sample Number	Input Images (t_{in} = 3, window size = 5-min)	Output Image
1	11:00, 11:05, 11:10	11:15
2	11:05, 11:10, 11:15	11:20
3	11:10, 11:15, 11:20	11:25
4	11:15, 11:20, 11:25	11:30
5	11:20, 11:25, 11:30	11:35
6	11:25, 11:30, 11:35	11:40
7	11:30, 11:35, 11:40	11:45
8	11:35, 11:40, 11:45	11:50
9	11:40, 11:45, 11:50	11:55
10	11:45, 11:50, 11:55	12:00
11	11:50, 11:55, 12:00	12:05
12	11:55, 12:00, 12:05	12:10
13	12:00, 12:05, 12:10	12:15

The red highlight indicates that taking these samples would result in an 'overlap' of the climate data at hours 11:00:00 and 12:00:00 and hence, these samples were discarded. Therefore, for any event, every 11th and 12th sample of rainfall images were discarded and thus, 10 samples were taken for every hour. This result also means that the climate data for every hour is duplicated 10 times. In the example of Table 19, the climate data at 11:00:00 is duplicated 10 times for the 10 samples i.e., 10 rows of 11:00:00 climate data.

Unlike the first, the second setup involves 'overlapping' the climate data. This was achieved by using t_{in} = 12 with 5-minute windows. Since the Keras ML library does not accept varying input shapes across samples during training as noted in 3.4.3, this issue was circumvented by taking a weighted moving average of the climate data (WMAC).

$$WMAC = \frac{n}{12}(Hour_i\ Climate\ Data) + \frac{12-n}{12}(Hour_{i+1}\ Climate\ Data)$$

Where n denotes the number of rainfall images within the i-th hour. For example, if the input images for t_{in} = 12 are 11:10 to 12:05, there are 10 rainfall images within the 11:00 hour and 2 rainfall images within the 12:00 hour. Three example samples are also depicted in Figure 20. These 2 setups are then compared to one another and the best performing setup will form the input setup for CNN_3 in all subsequent investigations.

Sample 1
Input Images: (11:00, 11:05, 11:10, 11:15, 11:20, 11:25, 11:30, 11:35, 11:40, 11:45, 11:50, 11:55)
Output Image: (12:00)

$$\therefore \textbf{WMAC} = 11{:}00\ Climate\ Data$$

Sample 2
Input Images: (11:05, 11:10, 11:15, 11:20, 11:25, 11:30, 11:35, 11:40, 11:45, 11:50, 11:55, 12:00)
Output Image: (12:05)

$$\therefore \textbf{WMAC} = \frac{11}{12} \times (11{:}00\ Climate\ Data) + \frac{1}{12} \times (12{:}00\ Climate\ Data)$$

Sample 3
Input Images: (11:10, 11:15, 11:20, 11:25, 11:30, 11:35, 11:40, 11:45, 11:50, 11:55, 12:00, 12:05)
Output Image: (12:10)

$$\therefore \textbf{WMAC} = \frac{10}{12} \times (11{:}00\ Climate\ Data) + \frac{2}{12} \times (12{:}00\ Climate\ Data)$$

$$\vdots$$

Figure 20 | Calculation of WMAC

3.4.7.3 Discretised Climate Data

Encoding the climate data into discrete categories was investigated to identify the effects, if any, on the model's performance. This is because the player's actions, which were categorical/discrete, was substituted for the climate data when adapting the video game model. To discretise the 6 climate variables, each variable was divided into 5 equal intervals. Each interval is then assigned an integer label from 0 to 4. For example, with z, the minimum and maximum values were 50,210.06 m^2s^{-2} and 57,035.18 m^2s^{-2} respectively. Therefore, z was discretised as follows:

Table 20 | Discretised intervals for geopotential height, z

Range [m^2s^{-2}]	Encoded Label
$50210.06 \leq z < 51580.54$	0
$51580.54 \leq z < 52944.20$	1
$52944.20 \leq z < 54307.86$	2
$54307.86 \leq z < 55671.52$	3
$55671.52 \leq z \leq 57035.18$	4

3.4.7.4 Spatial Aggregations

For the same reasons from 3.3.4.3, spatial aggregations with 5x5 squares was also investigated.

3.4.7.5 Combinations of Climatological Variables

Although 6 climatological variables were utilised in the baseline model, it was not apparent which of these 6 variables had the most effect in predicting high rainfall pixels. Additionally, the number of climatological variables utilised could also vary between 1 to 6. This meant that there were 63 possible combinations to investigate: ${}^{6}C_{1}+{}^{6}C_{2}+{}^{6}C_{3}+{}^{6}C_{4}+{}^{6}C_{5}+{}^{6}C_{6} = 63$, which implicates extensive computational times. Therefore, to minimise the number of investigations, the variables were grouped into 3 categories. By doing so, the number of possible combinations was reduced from 63 to 7. These are summarised in Tables 21 and 22.

Table 21 | 3 categories which the 6 variables were grouped into

Category	Climatological Variables
Atmospheric Properties (A)	Relative humidity (r) Temperature (t) Ozone mixing ratio (o_3)
Wind Speeds (W)	U-component of wind (u) V-component of wind (v)
Spatial Features (S)	Geopotential height (z)

Table 22 | Combinations of the 3 categories

Number of Categories	Included Categories
1	(1) A (2) W (3) S
2	(4) A + W (5) A + S (6) S + W
3	(7) A + W + S

4. Results and Discussion

4.1 Convolutional Neural Network 1

4.1.1 Transformations and Loss Functions

The evaluation results for all 6 different transformation and loss function combinations are shown in Table 23. A green colouring has been applied to visualise how the metrics vary between one another. The darker the colour, the better the performance. The best performing combination was chosen based on F for reasons stated in 3.3.2.

Table 23 | Results of the transformation and loss function combinations

Combination	Loss Function	Transformation	Recall (R)	Precision (P)	F1-Score (F)
1	MAE	Raw	18.95%	60.87%	28.90%
2		Transformation 1	9.79%	66.81%	17.07%
3		Transformation 2	12.92%	61.54%	21.36%
4	Logcosh	Raw	13.60%	68.79%	22.70%
5		Transformation 1	7.25%	65.48%	13.06%
6		Transformation 2	10.45%	53.25%	17.47%

The 2 best performing models are highlighted in white fonts. These models were trained on the raw data, which suggests that downscaling the rainfall intensities using transformations 1 and 2 results in poorer performances. A possible reason for this is attributed to how the intensities were weighted during training. Benshetler (2016) indicates that the smaller the range of inputs into a CNN, the more equally weighted the inputs are and vice versa. Since the aim of this project was concerned with the prediction of high rainfall intensities, it could be argued that higher rainfall intensities should be more heavily weighted i.e., assigned more importance in comparison to lower rainfall intensities to yield better predictions. This concept is also supported by comparing the evaluation metrics with the respective ranges of each transformation. Namely, the transformations that performed best to worst correspond to the raw data, transformation 2 and transformation 1 respectively. This arrangement also corresponds with the order from largest to smallest range: raw = 249.55 mm/hr, transformation 2 = 10.13 mm/hr and transformation 1 = 5.52 mm/hr. It can be inferred that higher rainfall intensities were assigned more importance using the raw data and thus, better predicted high rainfall pixels in comparison to the two transformation methods. Additionally, Gilon (2021) states that for CNNs, pixel intensities between [0, 255] usually yields the best results and the range of the raw rainfall intensities [0, 249.55] mm/hr best resembles this.

It is also observed that generally, the precision is significantly greater than the recall. The low recalls indicate that the model struggles to identify true extreme rainfall pixels. A possible explanation for this is attributed to the fact that the model was trained on a greater percentage of zero intensity pixels in comparison to higher intensity pixels. This is evidenced by the distribution of pixel intensities in Table 5. Hence, training the model on a high percentage of zero/low intensity pixels will enhance the accuracy of its predictions on these pixels. This argument is supported by the data from a full confusion matrix, such that the true negative rate (TNR) – the rate of correctly identifying non-extreme rainfall pixels is significantly greater than the recall. Nevertheless, since combination 1 performed the best, all subsequent investigations of CNN_1 was configured to this loss function and transformation combination.

Table 24 | Confusion Matrix for combination 1

		True Class [Y]		
		True [Y = 1]	False [Y = 0]	
Predicted Class [$\hat{Y}$]	True [$\hat{Y} = 1$]	TP = 1204	FP = 774	P = 60.87%
	False [$\hat{Y} = 0$]	FN = 5151	TN = 4338071	NPR = 99.88%
		R = 18.95%	TNR = 99.98%	

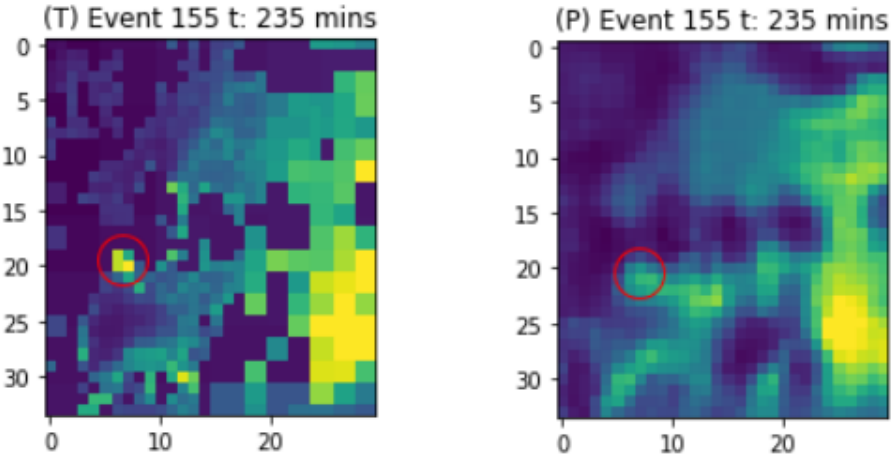


Figure 21 | True (left) and predicted (right) rainfall images for combination 1

Despite the low recall, the general spatial structure and extent of the rainfall areas are relatively well depicted. Additionally, the model is also able to correctly identify clustered pixels of high rainfall intensities, as seen towards the bottom right of each image. However, the predicted images are 'smoother' in comparison to its true counterpart, implying that the model is unable to detect abrupt spatial changes in pixel intensities. This is also shown by the distinct colouring gradient between pixels in the predicted image.

Furthermore, the model is also unable to identify localised high rainfall pixels, as circled in red. This is likely attributed to the fact that ML models thrive on identifying patterns, which must be relatively smoothly varying, and thus the model can identify clustered pixels of high rainfall intensities but not localised ones. Nevertheless, the qualitative analysis of this section demonstrates how quantitative analysis, while faster, can mislead one to assume that the model underperformed severely and possibly even discarding it when in fact, the figures show that there is some promise for further development as the model has generally depicted the overall spatial structure of the rainfall fields relatively well.

The qualitative observations described in this section were also verified by converting some of the output images into 2x2 spatially aggregated binary images. If any pixel within each 2x2 square exceeded the 40.84 mm/hr threshold, then that 2x2 square is classified as a high rainfall intensity area.

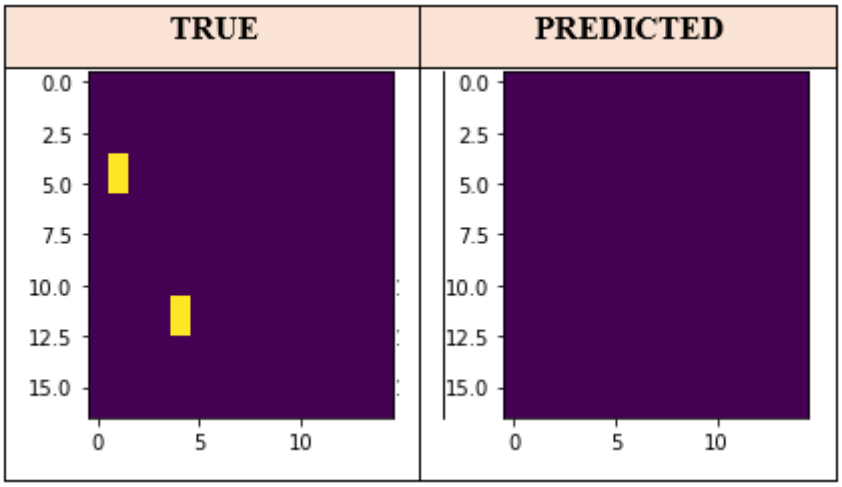


Figure 22 | Inability of model to identify localised pixels

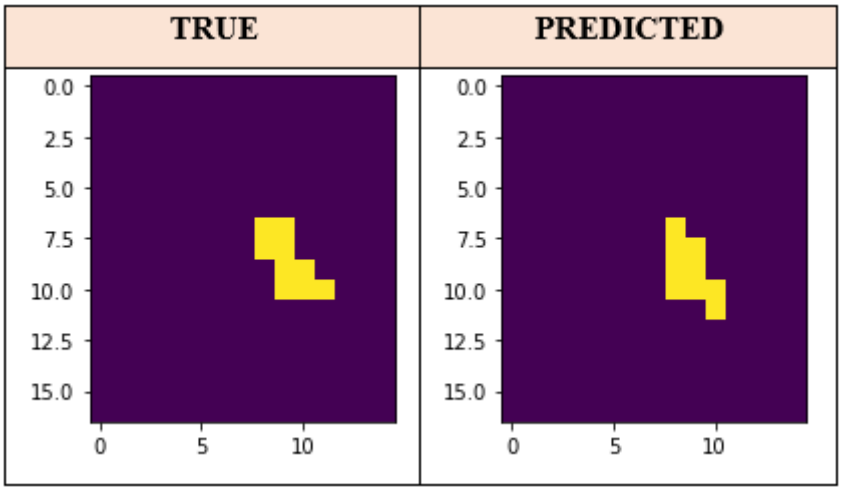


Figure 23 | Identified cluster of high rainfall intensities

4.1.2 Filter Sizes

Table 25 suggests that increasing the filter size (FS) improves the model's performance, albeit marginally. This is because increasing the filter size incorporates more parameters into the model, which enables the model to extract and learn more complex patterns and relationships from the input images during convolution. Although initially promising, the computational costs must also be accounted for. The results from Table 26 indicate that increasing the filter sizes represents a less than proportionate improvement in performance when factored against the increase in computational efforts. The green and red highlights indicate positive and negative features.

Table 25 | Quantitative results for FS = 3 and 5

Metric	Filter Size of 3	Filter Size of 5	% Change
Recall (R)	18.95%	19.67%	+3.8%
Precision (P)	60.87%	61.27%	+0.7%
F1-Score (F)	28.90%	29.78%	+3.0%

Table 26 | Parameter numbers and training times

Filter Size	$N_{PARAMETERS}$	Training Time per Epoch [s]
3	14,755	318
5	40,819	1248
% Change	+176.6%	+292.45%

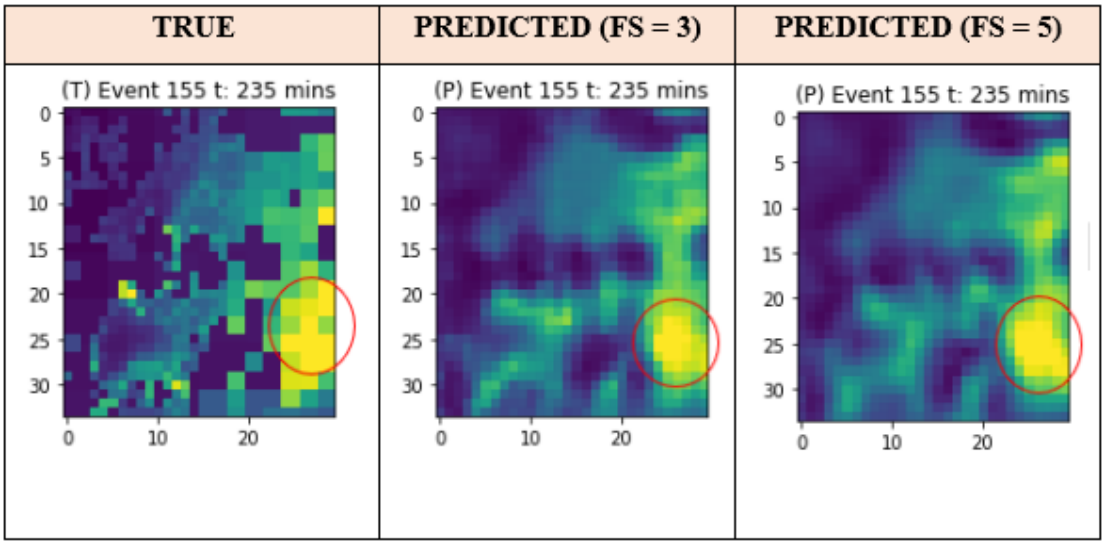


Figure 24 | Comparison between true and predicted rainfall images for FS = 3 and 5

From Figure 24, there were no significant qualitative differences by increasing FS from 3 to 5. It was also evident that despite the increased number of parameters, the inability of the model to predict localised pixels of high rainfall intensities persisted. However, the spatial extent of high rainfall clusters was better represented by the model with an increased FS, as circled in red, albeit slightly. Nevertheless, when factoring these marginal improvements against the computational efforts, increasing the filter size yields little to no benefit both quantitatively and qualitatively.

4.1.3 Spatial Aggregation

Increasing the spatial resolution to 5km x 5km lowers the 99.9th percentile threshold intensity to 25.9 mm/hr. This intensity still exists within the 'heavy' rainfall range defined by the UK Met Office (2011).

Table 27 | Quantitative comparison between 1km x 1km and 5km x 5km resolutions

Metric	1km x 1km	5km x 5km	% Change
Recall (R)	18.95%	51.06%	+169.4%
Precision (P)	60.87%	78.14%	+28.4%
F1-Score (F)	28.90%	61.76%	+113.7%

Table 28 | Pixel intensity distributions

Intensity Range [mm/hr]	% of Total Pixels		% Change
	1km x 1km	5km x 5km	
$x = 0$	54.90%	35.14%	-36.0%
$0 < x \leq 0.1$	7.37%	23.58%	+220.0%
$0.1 < x \leq 1$	22.89%	25.55%	+11.6%
$1 < x \leq 10$	14.46%	15.42%	+6.6%
$10 < x \leq 100$	0.37%	0.31%	-16.2%
$x > 100$	Negligible	Negligible	-

Table 27 suggests that spatially aggregating with 5x5 pixels improves the overall model performance significantly. To better understand why, the distribution of pixel intensities between the two spatial resolutions are compared. An observation made from Table 28 is the reduction in the proportion of zero intensity pixels. This reduction is attributed to the averaging of these pixels with higher intensity ones such that they are 'offset' from zero. Likewise, the reduction in the proportion of intensities between 10 - 100 mm/hr is attributed to spatially averaging these pixels with lower intensity ones. Hence, these reductions at the tail ends of the intensity distribution leads to increases in the middle intensity proportions i.e., a redistribution.

Since the proportion of zero intensity pixels was greater with a 1km x 1km resolution, the baseline model was therefore trained on a higher percentage of zero intensity pixels, thereby enhancing the model's forecasts on zero and low rainfall intensities, which explains its poor performance in correctly identifying extreme rainfall pixels. For example, if a model designed to classify images of cats and dogs was trained on a database where 99% of images were cats but only 1% were dogs, the model would therefore perform poorly when tasked with classifying an image of a dog from the test set. Under this reasoning, it can be predicted that F would increase as the threshold intensity decreases. This was verified by rerunning the spatially aggregated model with decreasing threshold intensities.

Table 29 | Performance metrics for decreasing threshold intensity values

Threshold [mm/hr]	Recall (R)	Precision (P)	F1-Score (F)
1	88.13%	91.59%	89.83%
10	63.65%	79.51%	70.70%
25.9	51.06%	78.14%	61.76%

Hence, given the net increase in the proportion of non-zero intensity pixels, the 5x5 spatially aggregated model is therefore trained on a greater percentage of non-zero intensity pixels relative to zero intensity pixels (64.86% vs 35.14%) in comparison to the baseline model (45.1% vs 54.9%). This aspect subsequently enables the spatially aggregated model to better predict non-zero rainfall intensities and hence, extreme rainfall pixels, thereby resulting in improvements across all 3 quantitative metrics.

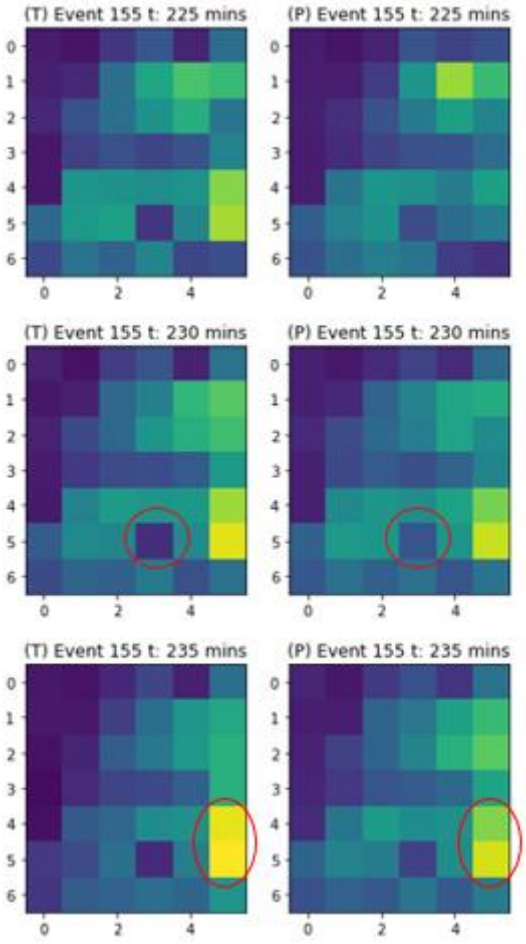


Figure 25 | Comparison between true (left) and predicted (right) images for 3 timesteps

Qualitatively, the spatial aggregations enabled the model to generate more accurate rainfall images with almost no discernible differences between the true and predicted images. This observation further verifies the quantitative comparison that the spatially aggregated model outperformed the baseline model. With the spatial aggregations, there were also no localised high rainfall pixels in the true rainfall images, unlike with a 1km x 1km resolution. This is because these localised pixels were averaged by its surrounding pixels and hence, eliminated. However, the spatially aggregated model still suffers from an inability to identify abrupt spatial changes between pixel intensities and therefore still produces 'smoother' images. These features are circled in red in Figure 25 where there is a distinct colouring gradient between pixels in the predicted images when compared to the true rainfall images. From a practical standpoint, this implies that the intensities of high rainfall pixels might be underpredicted whilst the intensities of low rainfall pixels might be overpredicted. Although spatially aggregating pixels might exhibit promise, one must exercise caution since doing so assumes the rainfall field is homogeneous within each 5km x 5km pixel, which is unlikely in practice.

4.1.4 Reduced Image Size

Table 30 | Quantitative comparison from training on 110x110 and 34x30 images

Metric	110x110	34x30	% Change
Recall (R)	18.95%	19.51%	+2.96%
Precision (P)	60.87%	60.70%	-0.28%
F1-Score (F)	28.90%	29.53%	+2.18%

Table 30 indicates a marginal improvement in the model's performance when trained on 34x30 images. This observation therefore contradicts the initial hypothesis postulated in 3.3.4.4. These improvements can likely be explained again by the reduction in the percentage of zero intensity pixels. However, the reduction in the proportion of zero intensity pixels from reducing the image sizes is likely a specific feature of this dataset as opposed to a general case of reducing image sizes.

Table 31 | Distribution of pixel intensities

Intensity Range [mm/hr]	% of Total Pixels		% Change
	110x110	34x30	
$x = 0$	54.90%	50.52%	-8.0%
$0 < x \leq 0.1$	7.37%	8.17%	+10.9%
$0.1 < x \leq 1$	22.89%	24.22%	+5.8%
$1 < x \leq 10$	14.46%	16.47%	+13.9%
$10 < x \leq 100$	0.37%	0.60%	+62.2%
$x > 100$	Negligible	Negligible	-

It can be inferred from Table 31 that the reduction and increases in the proportion of zero and non-zero intensity pixels respectively were the primary factors in improving the model's prediction of high rainfall pixels. However, the improvements with reduced image sizes is minute in comparison to the improvements observed with spatial aggregations. This is because the proportion of zero intensity pixels was reduced by a greater percentage when aggregating spatially in comparison to reducing the image size. Namely, the percentage change in the proportion of zero intensity pixels with spatial aggregations was -36% whereas reducing the image size only results in a percentage change of -8%.

Additionally, one might argue that the spatially aggregated model only performed better because it utilised a lower threshold intensity for classifying extreme rainfall pixels. This was falsified by rerunning the spatially aggregated model whilst using the same threshold intensity (40.84 mm/hr) as shown in Table 32.

Table 32 | 34x30 image model vs spatially aggregated model with the same thresholds

Metric	34x30	5km x 5km	% Difference
Recall (R)	19.51%	30.49%	+11.0%
Precision (P)	60.70%	52.08%	-8.6%
F1-Score (F)	29.53%	38.46%	+8.9%

Qualitatively, there were no discernible differences between the predicted images of the model when trained on the full image size (110x110) in comparison to when it was trained on the reduced image size (34x30). This observation therefore verifies the quantitative results that indicates only marginal improvements, which can be seen from Figure 26.

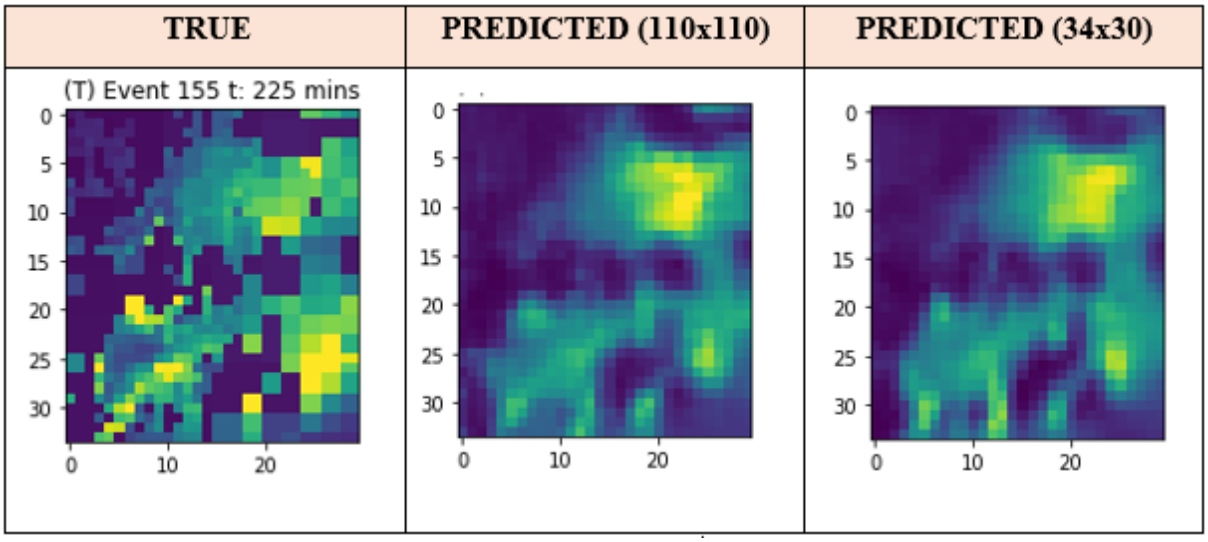


Figure 26 | Comparison of predicted images from model trained on full and reduced images

4.1.5 Increased t_{in}

Table 33 utilises the same colouring scale as Table 23. The best performing t_{in} for each individual metric was highlighted in white fonts.

Table 33 | Quantitative comparison of t_{in} = 2, 3 and 4

Metric	$t_{in} = 2$	$t_{in} = 3$	$t_{in} = 4$
Recall (R)	18.95%	21.50%	15.64%
Precision (P)	60.87%	61.68%	64.06%
F1-Score (F)	28.90%	31.89%	25.15%

Overall, t_{in} = 3 performed the best and is the most optimal input sequence for achieving the best predictions of high rainfall intensities. To explain why, it worth considering Tables 34 and 35 where the green and red colourings denote improvements and deteriorations in the performance metrics respectively.

Table 34 | Comparison between t_{in} = 2, 3

Metric	$t_{in} = 2$	$t_{in} = 3$	% Change
Recall (R)	18.95%	21.50%	+2.6%
Precision (P)	60.87%	61.68%	+0.8%
F1-Score (F)	28.90%	31.89%	+3.0%

Table 35 | Comparison between t_{in} = 3, 4

Metric	$t_{in} = 3$	$t_{in} = 4$	% Change
Recall (R)	21.50%	15.64%	-5.9%
Precision (P)	61.68%	64.06%	+2.4%
F1-Score (F)	31.89%	25.15%	-6.7%

Both Tables 34 and 35 indicate that increasing t_{in} from 2 to 3 improves the model's performance but increasing t_{in} from 3 to 4 deteriorates it. Hence, increasing t_{in} from 2 to 3 agrees with the initial prediction that increasing t_{in} would improve the model's predictions of high rainfall pixels whilst increasing t_{in} from 3 to 4 disagrees with the same prediction. These observations can be explained by the increased complexity of the input images. The main difference between using CNNs for the purposes of predicting high rainfall pixels and facial recognition is the dynamic nature of the input images.

Where facial images are static and independent with respect to time, rainfall images are not such that rainfall fields can change shape rapidly between 5-minute intervals, thus introducing increased complexities. Figure 28 exemplifies this rapidly changing shape over 25-minutes. Without sufficient input information to depict the dynamic evolution of rainfall fields, the model's performance may be hindered. Hence, by supplementing with additional images (i.e., 3 instead of 2), it enables the spatial features of the field to be better mapped and understood by the model as there is greater representation of their evolutions in more consecutive images. However, when t_{in} is increased from 3 to 4, confusion is likely introduced because these features will have moved too much, thereby increasing the amount of complexity in the input sequence. Therefore, these observations depict a trade-off between the additional information inputted and the degree of complexity introduced from increasing t_{in}.

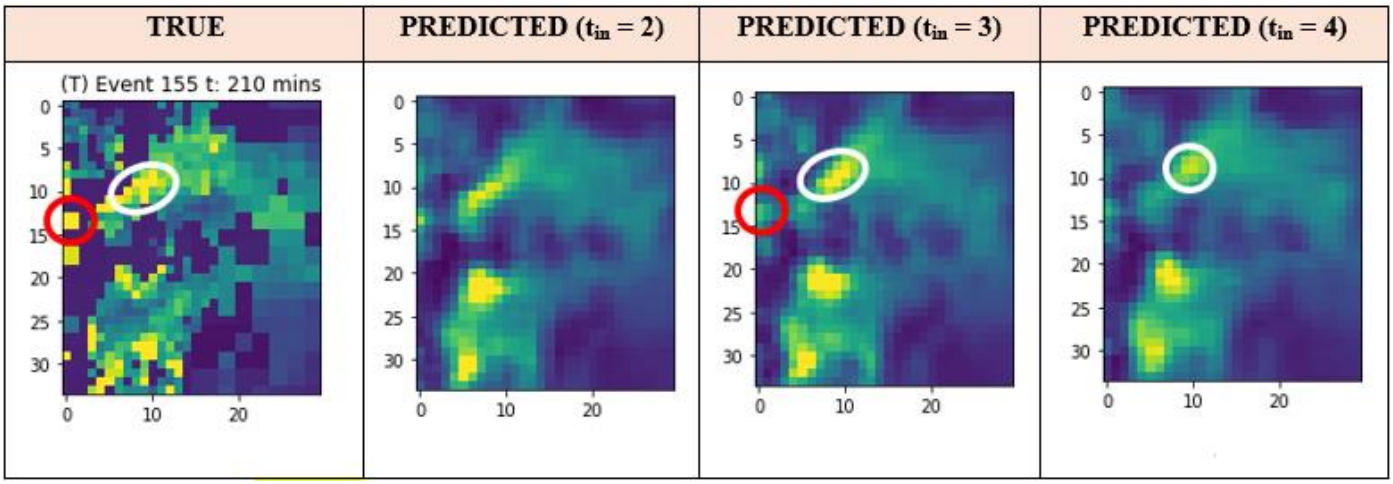


Figure 27 | Comparison between true and predicted images for t_{in} = 2, 3 and 4

Qualitatively, there were no significant differences between the predicted images for t_{in} = 2 and 3. However, for t_{in} = 4, there were visible deteriorations in its predictions as the spatial extent of some high rainfall intensity clusters (circled in white) has reduced in comparison to t_{in} = 2 and 3. This observation therefore aligns with the quantitative results that the model's performance falters if t_{in} is increased beyond 3. In general, all t_{in} setups of the model were able to correctly identify the general spatial structure of the rainfall field.

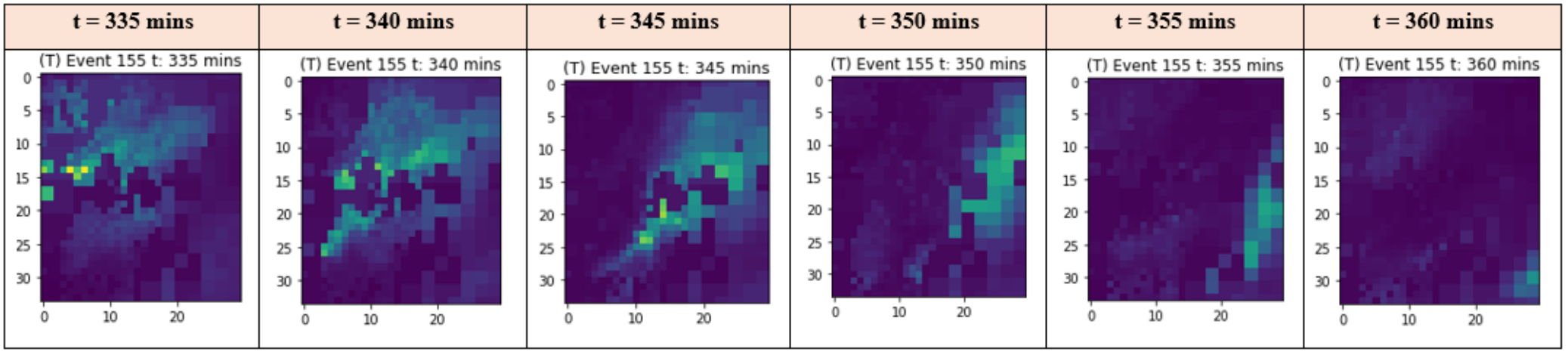


Figure 28 | Evolution of rainfall field over 25-minutes for Event 155

4.1.6 Timestep Irregularity

Table 36 | Quantitative comparison for t_{in} = 2: Sequential [S] and non-sequential inputs

Metric	t_{in} = 2 [S]	t_{in} = 2 [P_1]	% Change
Recall (R)	18.95%	8.65%	-10.3%
Precision (P)	60.87%	59.69%	-1.2%
F1-Score (F)	28.90%	15.11%	-13.8%

Table 37 | Quantitative comparison for t_{in} = 3: Sequential [S] and non-sequential inputs

Metric	t_{in} = 3 [S]	t_{in} = 3 [P_2]	t_{in} = 3 [P_3]
Recall (R)	21.50%	6.56%	17.25%
Precision (P)	61.68%	63.41%	62.72%
F1-Score (F)	31.89%	13.51%	27.05%

Tables 36 and 37 suggest that timestep irregularity deteriorates the model's predictive capabilities significantly. Hence, the model performs best when the input images are sequential, and the temporal resolution is maintained between all input images. These observations can be understood by the explanation made in 4.1.5 such that when timesteps are missing in between, the rainfall structure would have 'progressed' more. For example, the rainfall field is more likely to have travelled a greater distance and changed shape over a 10-minute interval than a 5-minute interval. By removing timesteps in between, there are key 'fragments' of information on the evolution of the rainfall structure missing from the input sequence. For example, from Figure 28 and taking P_2 as the input sequence, this would be the equivalent of using t = 340, 345 and 355 mins to predict t = 360 mins. However, the dissimilarities between the input images at 345 mins and 355 mins in tandem with the lack of information from t = 350 mins means it is more obscured to the model how this rainfall field has evolved, which results in a deteriorating performance.

For t_{in} = 3, the model performed better with P_3 than it did with P_2 as seen in Table 37. The difference between the two sequences is the consecutiveness of the inputs relative to the output image. With P_2, this was only t+15 whilst with P_3, these were t+10 and t+15. Hence, P_2 had 1 consecutive input image relative to predicting t+20 whereas P_3 had 2. This suggests that the more consecutive the input images are relative to the output image, the better the model's performance in predicting high rainfall pixels. This is because the combination of 2 consecutive images provides more information on the progression and movement of the output rainfall field in comparison to just 1 image.

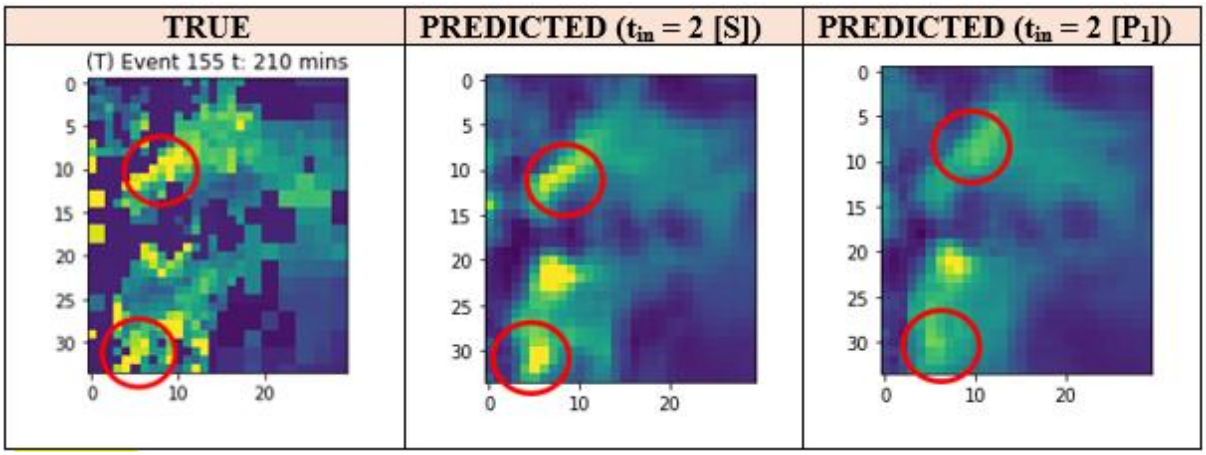


Figure 29 | Comparison of predicted images for sequential and non-sequential inputs (t_{in} = 2)

The qualitative observations from Figure 29 align with the quantitative results. Most notably, the spatial extent of high rainfall pixels as circled in red were not captured as well for non-sequential inputs in comparison to sequential inputs. However, the general shape of the rainfall field was still discernible in the predicted images of non-sequential inputs. Therefore, these observations suggest that the effect of timestep irregularity has greater impact on the prediction of high rainfall pixels as opposed to the general shape of the rainfall field itself.

4.1.7 Increased Forecast Lead Time

Table 38 | Quantitative comparison of the 5, 10, 15-min lead times for t_{in} = 2

Metric	5-min Lead	10-min Lead	15-min Lead
Recall (R)	18.95%	0.44%	0%
Precision (P)	60.87%	51.85%	0%
F1-Score (F)	28.90%	0.88%	0%

Table 38 indicates that the model's ability to identify high rainfall pixels deteriorates significantly as the lead time increases. This suggests that the ability of the model to predict high rainfall pixels is inversely proportional to the lead time. This is due to the mismatch between the temporal resolution of the input sequence and the lead time. Namely, the temporal resolution of the input sequence is 5-minutes but the increased lead times of 10 and 15-minutes means that there is a lack of information from more recent images in between the last input image and the output image. For example, from Figure 28 and taking a lead time of 15-minutes, this would be the equivalent of using the images at t = 340 and 345 mins as inputs whilst skipping over t = 350 and 355 mins to predict t = 360 mins. Thus, the most recent pieces of information on the movements and evolution of the spatial structure of the rainfall field are not utilised, which subsequently hinders the model's performance.

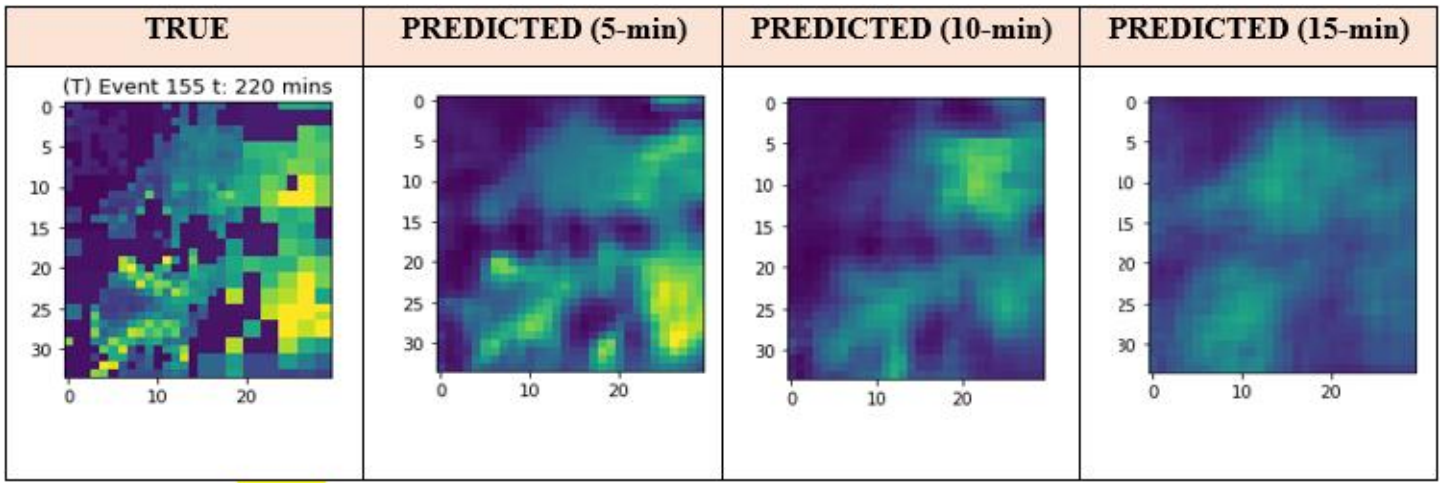


Figure 30 | Comparison of predicted images for 5, 10 and 15-minute lead times

Figure 30 verifies the quantitative analysis that no high rainfall pixels were predicted by the model with a 15-minute lead time and that there is a clear deterioration in model performance as the lead time increases. This is shown by the lack of bright coloured pixels in the predicted images as the lead time increases. Additionally, the general shape and extent of the rainfall field is more distorted in comparison to the predicted image for a 5-minute lead time. From an operational perspective, this has negative implications as it means areas of high rainfall are identified less accurately or not at all and the spatial structure of the rainfall field cannot be approximated as the lead time increases.

4.1.8 Increased Temporal Resolution

Table 39 | Quantitative comparison of 5-min and 15-min temporal resolutions for $t_{in} = 2$

Metric	5-min res.	15-min res.	% Change
Recall (R)	18.95%	0%	-100%
Precision (P)	60.87%	0%	-100%
F1-Score (F)	28.90%	0%	-100%

Table 39 suggests that increasing the temporal resolution negatively impacts the model's performance. A plausible reason for this is due to the number of samples available for training. Most notably, by increasing the temporal resolution to 15-minutes, the number of training samples has decreased by a factor of 3. These observations pertain to the issue of negatively impacting the model's performance by reducing the number of training samples. Brownlee (2019d) states that more training samples will provide the training algorithm with more opportunities to better understand and hence, more accurately map the underlying relationship between input and output. This in turn leads to a better performing model. With a temporal resolution of 15-minutes, the training algorithm has 3 times as fewer opportunities to better understand and map the relationship between input and output in comparison to a 5-minute resolution model. Therefore, without additional samples, increasing the temporal resolution will result in the model performing worse relative to a model trained on the minimum/finest temporal resolution.

Table 40 | Sample sizes for 5-minute and 15-minute temporal resolutions

t_{in}	Temporal Resolution	Total Samples	Training Samples
2	5-minutes	28,409	19,886
	15-minutes	9,343	6,540

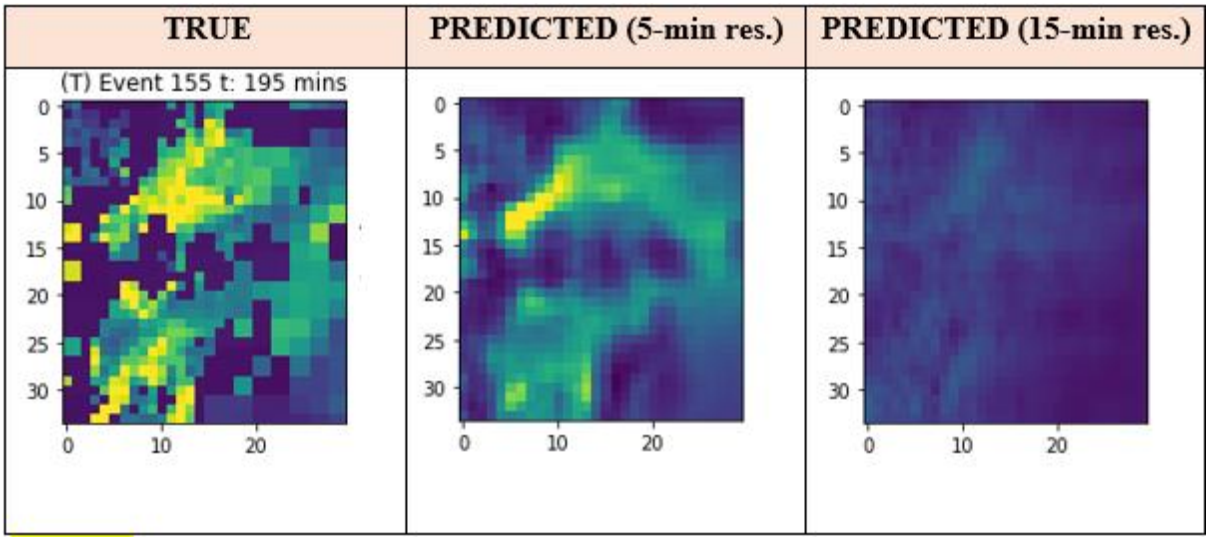


Figure 31 | Comparison of predicted images for 5-minute and 15-minute temporal resolutions

Qualitatively, the 15-minute resolution model performed significantly worse relative to its 5-minute resolution counterpart. The general shape of rainfall field could also no longer be approximated by the 15-minute resolution model. Furthermore, the lack of bright coloured pixels verifies the quantitative assessment that there were no hits on high rainfall pixels.

4.2 Convolutional Neural Network 2

4.2.1 Results

Table 41 | Performance of CNN_2 on the combinations of loss function and transformation

Combination	Loss Function	Transformation	Recall (R)	Precision (P)	F1-Score (F)
1	MAE	Raw	0%	0%	0%
2		Transformation 1	0%	0%	0%
3		Transformation 2	0%	0%	0%
4	Logcosh	Raw	0%	0%	0%
5		Transformation 1	0%	0%	0%
6		Transformation 2	0%	0%	0%

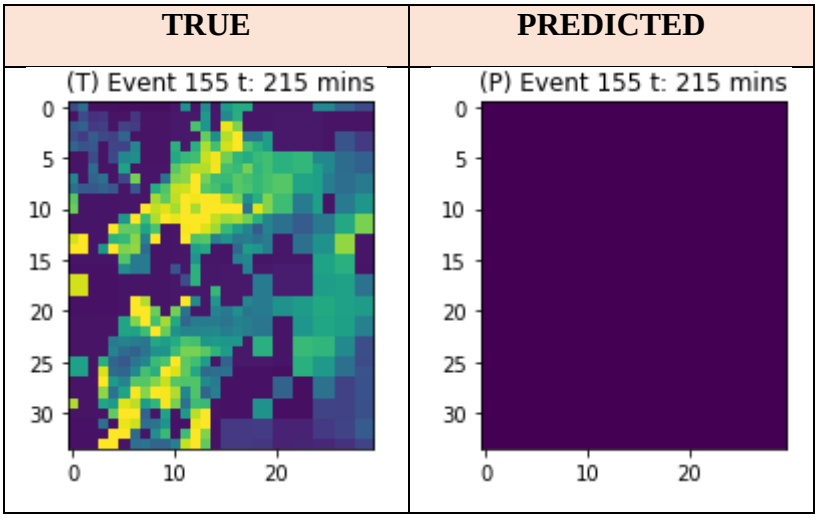


Figure 32 | Comparison between true and predicted images for CNN_2 (combination 1)

The most interesting observation is the zero-hit rate across all combinations, which was also verified qualitatively by Figure 32. These results therefore suggest that the model is unable to identify any high rainfall pixels. To pinpoint the cause for the zero hit rates on high rainfall pixels, it was hypothesised that there was a fundamental issue with the architecture of CNN_2. This is because the basis for convolving the input images with valid padding followed by flattening into fully connected layers was to combine the input images with climatological data in CNN_3, which the architecture of CNN_2 is identical to but excludes the climate branch of the network (compare Figures 12 and 13). Hence, it could be argued that this climate branch was integral towards obtaining suitable predictions and without inputs from this branch, the architecture of CNN_2 is fundamentally 'flawed.' To verify this hypothesis, a repeating sequence of integers from 1 to 6 was inputted as 'useless' climate data through the climate branch of CNN_3. Thus, if the results indicated that the model performance improved even with 'useless' climate data, it would therefore verify the idea that the climate branch is integral. Table 42 summarises these findings.

Table 42 | Summary of evaluation metrics

Metric	CNN_2	CNN_3 (w/ 'useless' climate data)
Recall (R)	0%	0%
Precision (P)	0%	0%
F1-Score (F)	0%	0%

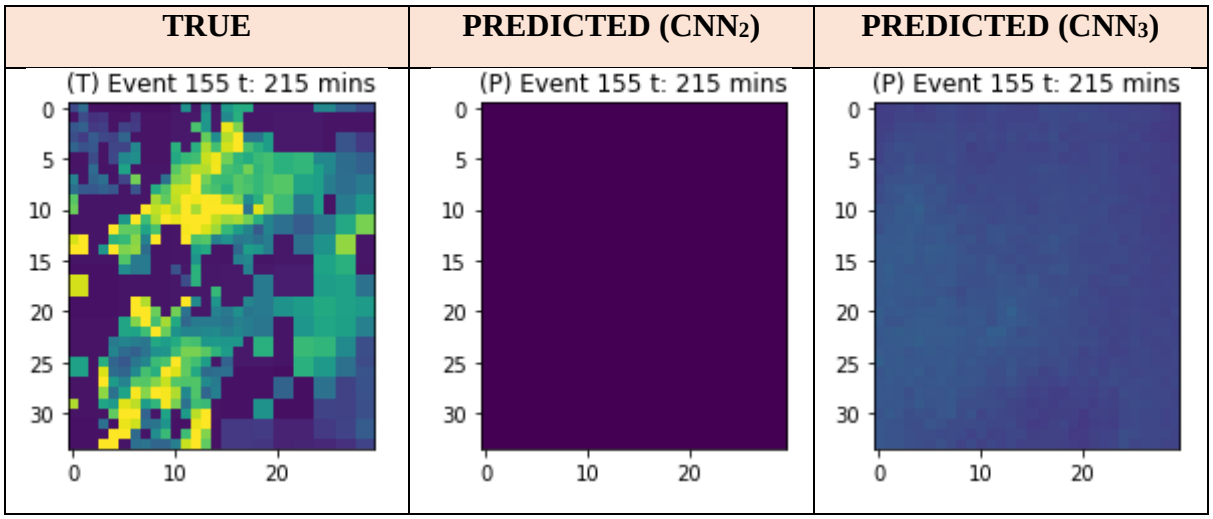


Figure 33 | Comparison of true and predicted images for CNN_2 and CNN_3 (with 'useless' climate data inputted)

When considering Table 42 alone, the results suggest that the hypothesis is wrong and that the climate branch is unimportant since no performance improvements were quantifiable. However, qualitative analysis of Figure 33 suggests otherwise. Although the shape of the rainfall field is not discernible in the predicted image of CNN_3, it is apparent that the prediction of rainfall intensities has improved. This is evidenced by the brighter pixels in the predicted image of CNN_3 in comparison to the darker pixels of CNN_2. In fact, the maximum intensity predicted by CNN_3 even with 'useless' climate data is 24.5 mm/hr, which is significantly greater than the < 1 mm/hr maximum predicted by CNN_2. Thus, the fact that the model was able to predict higher rainfall intensities even with 'useless' climate data arguably bears greater resemblance to the true image and therefore exhibits more promise. These findings subsequently validate the hypothesis that the climate branch is integral and without inputs from it, the architecture of CNN_2 is fundamentally 'flawed.' Hence, it is not worthwhile investigating CNN_2 further given these findings.

4.3 Convolutional Neural Network 3

4.3.1 Transformations and Loss Functions

Table 43 | Quantitative results from the 18 combinations investigated on the baseline model

Combination	Loss Function	Rainfall Transformation	Climate Transformation	R	P	F
1	MAE	Raw	Raw	2.43%	0.10%	0.19%
2			Normalised	0%	0%	0%
3			Equalised	0.20%	0.15%	0.17%
4		Transformation 1	Raw	2.85%	0.21%	0.40%
5			Normalised	0%	0%	0%
6			Equalised	0%	0%	0%
7		Transformation 2	Raw	0%	0%	0%
8			Normalised	0%	0%	0%
9			Equalised	0%	0%	0%
10	Logcosh	Raw	Raw	10.75%	0.68%	1.28%
11			Normalised	0%	0%	0%
12			Equalised	0%	0%	0%
13		Transformation 1	Raw	0%	0%	0%
14			Normalised	0%	0%	0%
15			Equalised	0%	0%	0%
16		Transformation 2	Raw	0%	0%	0%
17			Normalised	0%	0%	0%
18			Equalised	0%	0%	0%

The results indicate that overall, the model performed poorly with regards to predicting high rainfall pixels. However, it is worth recalling that since 1-hour windows with $t_{in} = 12$ was used, the number of training samples was significantly reduced, which was necessary to reduce computational times. Hence, it was predicted that the model's performance would be compromised. Therefore, the goal of this investigation was not to identify which combination performed 'good,' but rather, which combination exhibited the most promise even with a reduced sample size. From Table 43, one might be inclined to choose combination 10 since it exhibited the highest recall, precision, and F1-score. However, qualitative analysis of the predicted images suggests otherwise. This was because it was found that a series of 'erroneous' images was predicted by some combinations, which is depicted in Figure 34.

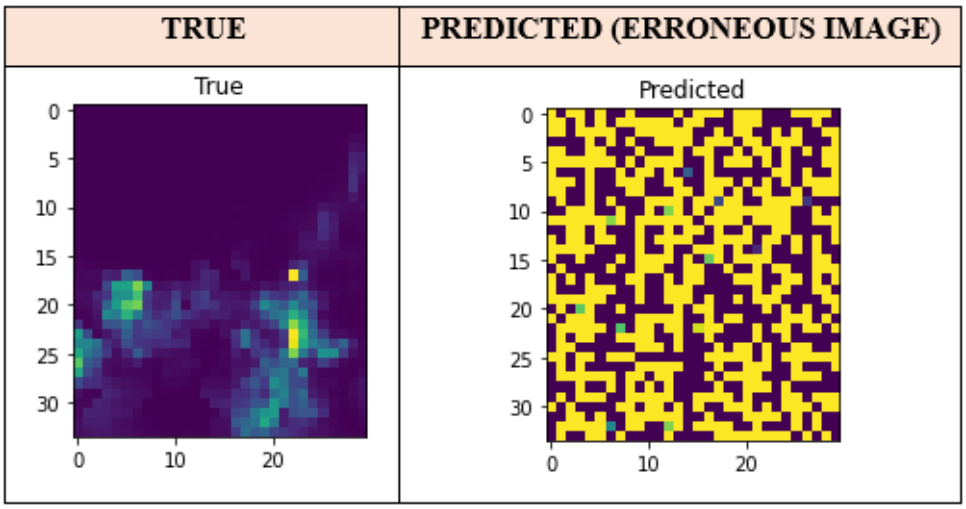


Figure 34 | Erroneous predicted image

Consequently, this prompted identification of the combinations that outputted these erroneous images so that they could be eliminated. This was achieved by qualitatively analysing the predicted images of each combination and tallying the number of erroneous images. Table 44 summarises these tallies.

Table 44 | Number of erroneous images for each combination

Combination	Loss Function	Rainfall Transformation	Climate Transformation	$N_{ERROR\ IMAGES}$
1	MAE	Raw	Raw	26
2			Normalised	0
3			Equalised	14
4		Transformation 1	Raw	12
5			Normalised	0
6			Equalised	0
7		Transformation 2	Raw	10
8			Normalised	0
9			Equalised	0
10	Logcosh	Raw	Raw	27
11			Normalised	0
12			Equalised	19
13		Transformation 1	Raw	7
14			Normalised	0
15			Equalised	0
16		Transformation 2	Raw	5
17			Normalised	0
18			Equalised	0

It is interesting to note that erroneous images are observed for every combination that had a non-zero hit rate on high rainfall pixels. This relationship indicates that these combinations were only able to predict high rainfall pixels due to the erroneous images. Hence, these combinations were removed from consideration. It can also be seen from Table 44 that erroneous images are generated every time the climate data is in its raw form. This is attributed to the fact that when the climate data is unscaled, the scales between climate variables vary significantly. For example, the geopotential height (z) ranges between [50210.06, 57035.18] m^2s^{-2} whilst the ozone mixing ratio (o_3) varies between [5.36×10^{-8}, 2.03×10^{-7}]. When these unscaled climate variables are inputted into the model, each variable is therefore weighted differently during training in accordance with their scales. For example, the large scale of z associates it with large weights whilst the small scale of o_3 associates it with small weights during training. In effect, the varying weights means that each climate variable is not treated with equal importance during the training process (i.e., biases are introduced) and Brownlee (2019e) states that these biases can lead to an unstable training process, which in turn results in higher prediction errors. Another notable observation from Table 44 is the generation of erroneous images when the climate variables were equalised to the same range as the raw rainfall intensities but not when the intensities were transformed.

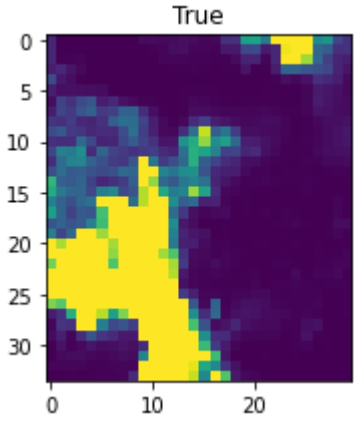
These observations are attributed to the fact that the scale of each climate variable is considered large, despite equally weighting all variables by equalising them to the same range as the raw rainfall intensities. This is because Brownlee (2019e) highlights that another factor in addition to biases that results in high prediction errors are large weights, which is associated with large input scales - hundreds of units or more. Hence, when the range of all 6 climate variables were equalised to the same range as the transformed rainfall intensities: [0, 5.52] and [-4.61, 5.52], the scales were made ‘small’ and thus, the issue of large weights was circumvented, thereby eliminating the observation of erroneous images (i.e., equally weighted with ‘small’ scales). These explanations are further validated by the fact that no erroneous images were observed when the climate variables were normalised to [0, 1]. Nevertheless, if there was at least 1 erroneous image observed for any combination, then that combination was eliminated. This meant combinations 1, 3, 4, 7, 10, 12, 13, and 16 were removed from consideration.

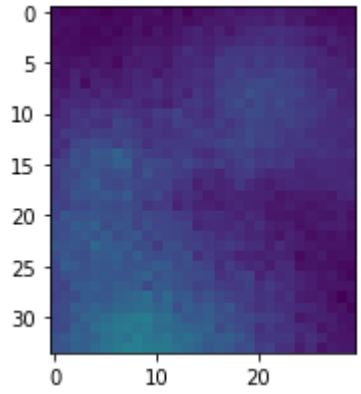
Next, the maximum predicted intensity of each combination was compared, and if the maximum predicted intensity of any combination was < 3 mm/hr, then these combinations would be removed. The maximum predicted intensity was compared because it immediately identifies which combinations would only predict low rainfall intensities.

Table 45 | Maximum intensities for each combination (red denotes maximum intensities < 3 mm/hr)

Combination	Loss Function	Rainfall Transformation	Climate Transformation	Max. Intensity [mm/hr]
2	MAE	Raw	Normalised	18.40
5		Transformation 1	Normalised	0.30
6			Equalised	2.37
8		Transformation 2	Normalised	54.20
9			Equalised	2.48
11	Logcosh	Raw	Normalised	19.76
14		Transformation 1	Normalised	11.39
15			Equalised	2.65
17		Transformation 2	Normalised	17.83
18			Equalised	2.63

Under this criterion, combinations 5, 6, 9, 15 and 18 were discarded. From the remaining combinations, qualitative analysis was conducted to identify the combinations that best predicted the shape of the rainfall field and distribution of high rainfall pixels. Figure 35 exemplifies this next step.



COMBINATION	PREDICTED IMAGE
2	

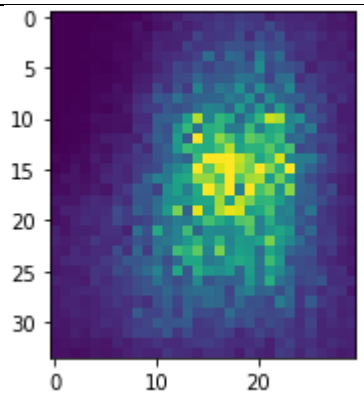
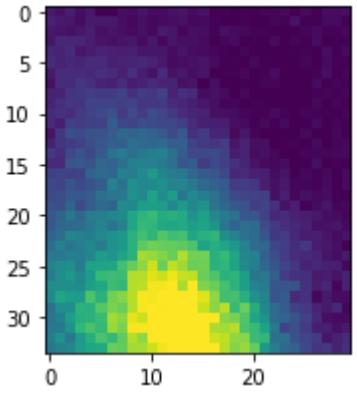
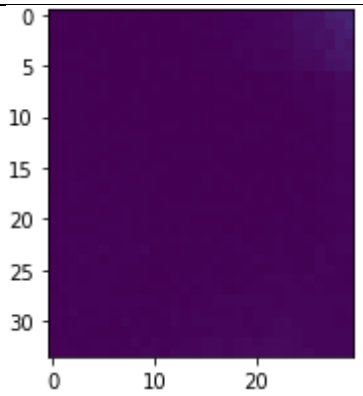
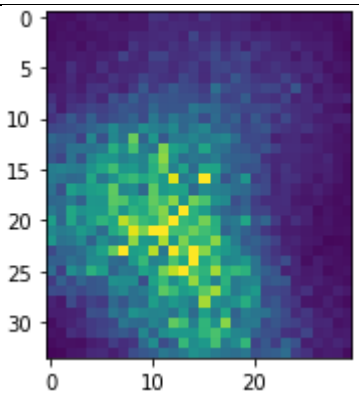
8	
11	
14	
17	

Figure 35 | Comparison of the predicted images for the remaining combinations

From Figure 35, combinations 8, 11 and 17 were the best predictors of high rainfall pixels. This was evidenced by the presence of bright pixels in the predicted images. It was also observed that combination 17 was the best at correctly predicting the shape of the general rainfall field. However, combination 11 outperformed combinations 8 and 17 when predicting the spatial distribution of high rainfall pixels since it was the only one that correctly identified the cluster of high rainfall pixels and its general position, albeit not to the same spatial extent. It could be reasoned that predicting the spatial distributions of high rainfall pixels is more important for flood warning purposes and under this reasoning, combination 11 exhibits the most promise. Hence, all subsequent investigations of CNN_3 was configured to combination 11.

4.3.2 Varying Input Setup (t_{in} and Window Size)

Since the most promising combination of loss function, rainfall, and climatological transformation had been identified, the sample size was then increased by varying t_{in} and the window size to quantify any improvements in the model's performance and to subsequently identify an appropriate input setup for CNN_3.

Table 46 | Results from varying t_{in} and window size to increase the sample size

Setup	t_{in}	Window Size	Samples	Recall (R)	Precision (P)	F1-Score (F)
1 (No overlap)	3	5-minutes	22,580	0.64%	40.51%	1.26%
2 (Overlap)	12	5-minutes	25,699	0.15%	26.67%	0.31%

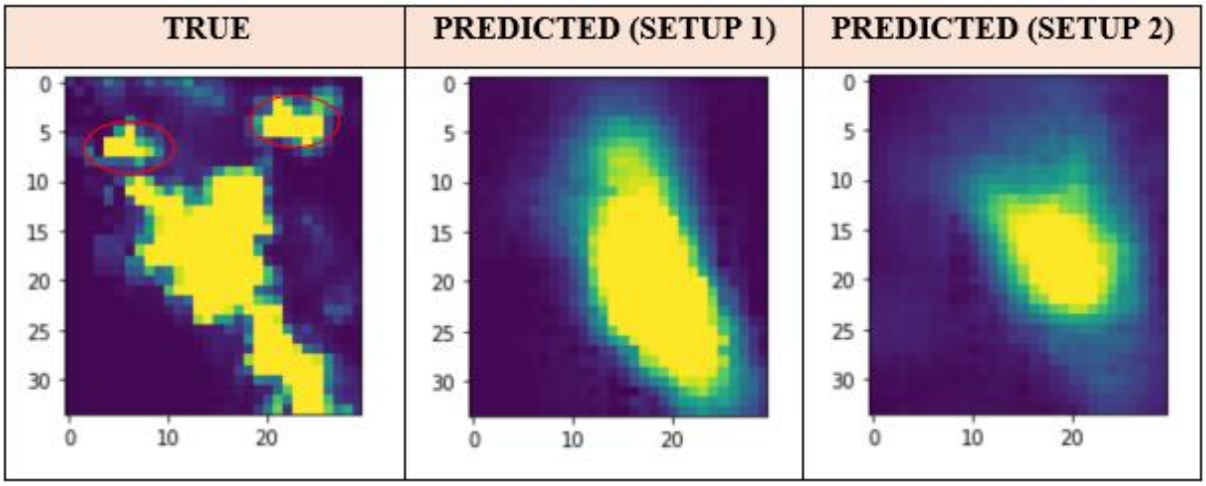


Figure 36 | Comparison of predicted images for setups 1 and 2

From qualitative analysis of the 2 setups, Figure 36 verifies the quantitative results that setup 1 outperformed setup 2. This is because while both setups identified the position of the high rainfall cluster to a satisfactory degree, setup 1 better predicted the spatial extent of this cluster in comparison to setup 2. However, in general, both setups failed to identify the presence of smaller and more localised clusters, as circled in red. Additionally, when high rainfall pixels accounted for a small portion of the rainfall image, the model struggled to identify any of these pixels in both setups. This is shown in Figure 37. Overall, although the metrics are indicative of poor predictive capabilities, the predicted images suggest that there is some promise in predicting high rainfall pixels since it was able to identify the larger clusters.

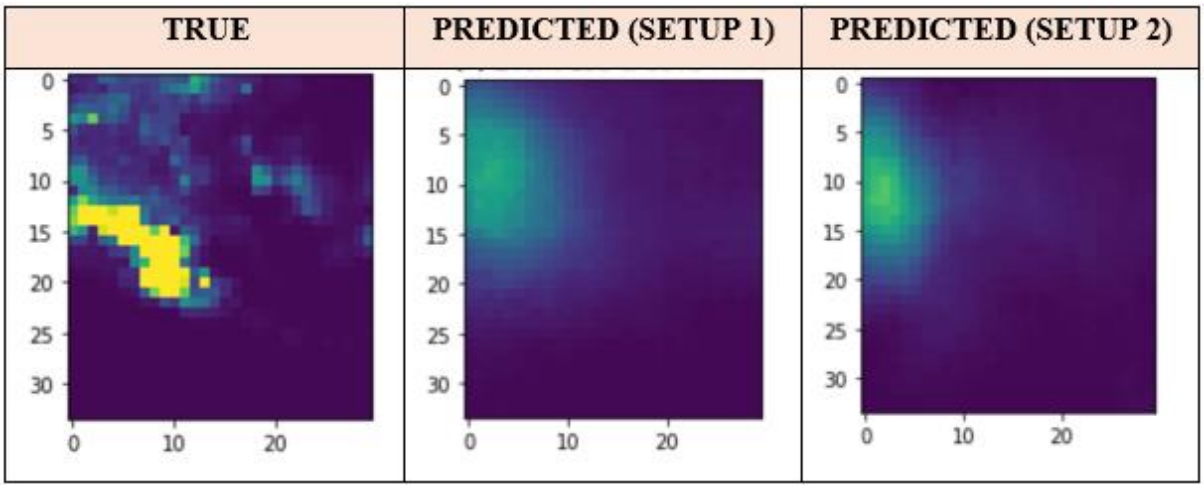


Figure 37 | CNN_3 struggling to identify smaller high rainfall clusters regardless of setup

It should be noted however, that both setups have their limitations. For example, by overlapping the climate data, linearity is assumed between consecutive timesteps by taking the weighted averages due to software related reasons stated in 3.4.7.2, which is arguably unrealistic and inaccurate. On the other hand, with no overlaps (i.e., setup 1), it was assumed that the hourly climate data was applicable to every 5-minute interval within each hour such that each hourly climate data could be duplicated 10 times to match the number of rainfall image samples, which was also highlighted in 3.4.7.2. Doing so implies that the climatological features are static across all 5-minute intervals within each hour, which is also unrealistic and inaccurate. Therefore, although these setups were necessitated by the unequal temporal resolutions of the rainfall and climate data, the climate data might not have had its true effects accurately simulated by the model. This issue is verified in section 5. Nevertheless, since the goal of this investigation was to identify the best performing input setup and both the quantitative and qualitative results favour setup 1, setup 1 is therefore chosen as the new input setup for CNN_3 for all subsequent investigations.

4.3.3 Discretised Climate Data

Table 47 | CNN_3 performance with and without discretised climate data

Climate Data	Recall (R)	Precision (P)	F1-Score (F)
Numerical	0.64%	40.51%	1.26%
Discretised	0%	0%	0%

Table 47 indicates that the model performs poorly with discretised climate data. To explain these results, it could be argued that discretising the climate data is fundamentally incorrect and needless whereas encoding the player's actions into discrete numerical labels was necessary. This is due to the nature of the climate data in comparison to a player's actions. Namely, the climate data are continuous numerical variables and therefore does not require discretisation into intervals as shown in 3.4.7.3. On the other hand, a player's actions are categorical in the video game and hence requires encoding into discrete numerical labels since ML models do not accept raw categorical data as inputs. Therefore, by discretising the climate data into intervals and subsequently assigning numerical labels to these intervals, the properties of each climatological variable such as the distribution is fundamentally altered. This is exemplified by Figure 38, which compares the scatter plots of the relative humidity over all hourly time points before and after discretisation.

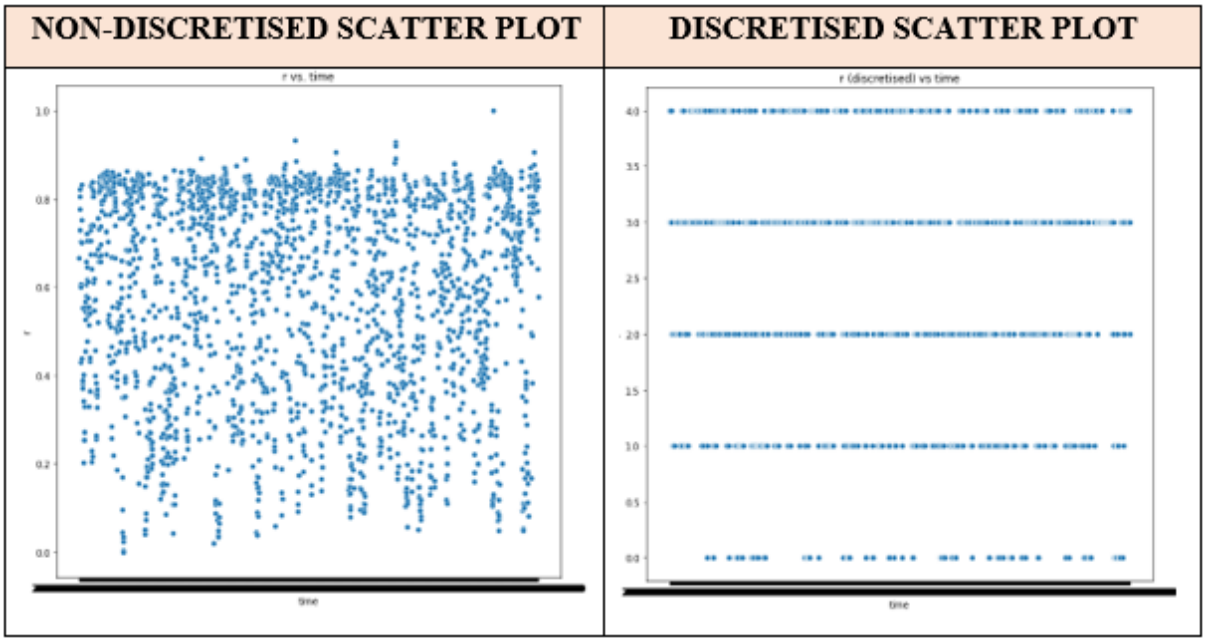


Figure 38 | Relative humidity (r) vs time scatter plots before and after discretisation

Hence, by discretising the climate data, its discretised formats are incorrect representations of the climate data in its original numerical form. Thus, it can be reasoned that the climate data is not the ‘same’ after it has been discretised and therefore any underlying statistical relationships between the climatological features and high rainfall intensities are subsequently invalidated by discretisation. These observations were also supported by qualitative analysis of the predicted images in Figure 39 such that there was an absence of bright coloured pixels, which therefore verifies the zero-hit rate on high rainfall pixels.

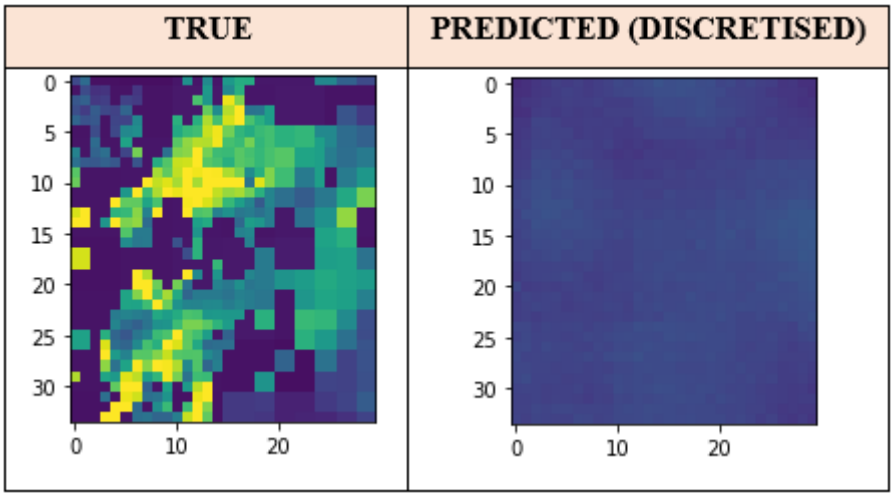


Figure 39 | Comparison of true vs predicted images using discretised climate data

4.3.4 Spatial Aggregations

Table 48 | Performances with varied spatial resolutions

Spatial Resolution	Recall (R)	Precision (P)	F1-Score (F)
1km x 1km	0.64%	40.51%	1.26%
5km x 5km	0%	0%	0%

Table 48 shows that the spatially aggregated model of CNN_3 was unable to identify any high rainfall pixels and thus the non-spatially aggregated model performed better by comparison. A possible reason for these zero hit rates in the spatially aggregated model is attributed to its architecture, which had to be altered slightly to be trained on spatially aggregated pixels. Taking 5x5 spatial aggregations means that the input images are reduced to 22x22. The consequence of this is in tandem with valid padding means that the input image size would constantly decrease as it is fed through subsequent convolutional layers. This is because for an image of height H and width W, after it has been convolved in a convolutional layer with valid padding, the convolved feature height H' and width W' are given by:

$$H' = \frac{H - FS}{S} + 1$$

$$W' = \frac{W - FS}{S} + 1$$

Where FS and S denote the filter size and stride respectively. Hence, to avoid the issue of a dimensionless convolved feature, some convolutional and deconvolutional layers had to be removed. This meant that layers CONV2, CONV3, DCO1 and DCO3 were discarded. Consequently, it could be argued the removal of these convolutional and deconvolutional layers from a model that was already struggling to predict high rainfall pixels might have flawed the model architecture and thus important features from each input image (e.g., the presence of high rainfall pixels, the spatial distribution of these pixels, etc.) were not extracted. As a result, this led to the model's inability to predict high rainfall pixels. These quantitative observations were also verified by the lack of bright coloured pixels in the predicted images as exemplified in Figure 40.

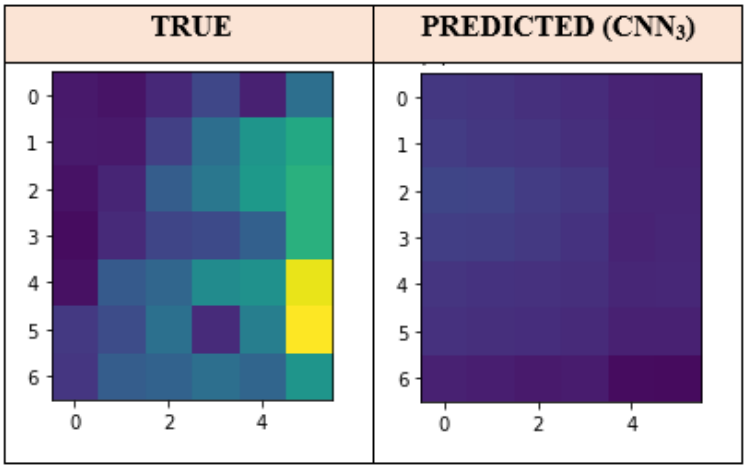


Figure 40 | Comparison between true and predicted images

4.3.5 Combinations of Climatological Categories

Table 49 | Results for different combinations of climate variables of CNN_3 ranked by F

Rank	Combination	Recall (R)	Precision (P)	F1-Score (F)
1	(6) S + W	1.36%	59.65%	2.66%
2	(3) S	0.78%	40.21%	1.53%
3	(2) W	0.64%	48.48%	1.26%
4	(7) A + W + S	0.64%	40.51%	1.26%
5	(5) A + S	0.38%	67.86%	0.75%
6	(4) A + W	0%	0%	0%
6	(1) A	0%	0%	0%

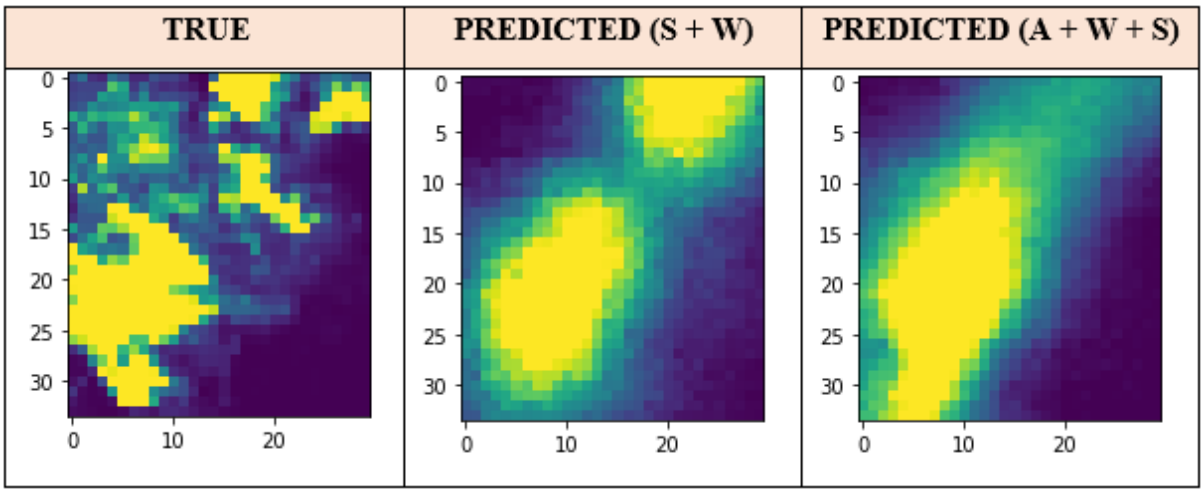


Figure 41 | Comparison of predicted images from combinations (6) and (7)

It should be noted that with combination (7), this corresponds to setup 1 of CNN_3 in 4.3.2 where all 6 climatological variables were included in the training process. Although the quantitative metrics indicate that the model performed poorly regardless of combination, qualitative analysis suggests otherwise. For example, from Figure 41, the model is generally able to identify high rainfall clusters and its positions for both combinations. However, qualitative analysis also agrees with some of the quantitative results. For example, the predicted image of combination (6) could be argued to better resemble the true image than combination (7).

This is because the former was able to identify the localised clusters towards the top right corner, which the latter was incapable of emulating. Hence, this observation agrees with the rankings of Table 49. Nevertheless, both combinations also exhibit some similarities such as overpredicting the spatial extent and depicting a more 'oval' shaped cluster in comparison to the true images. Furthermore, the results from Table 49 depicts a discernible relationship between the different climate categories incorporated into CNN_3. Namely, the best performing combinations included S and/or W whilst the worst performing combinations always included A. To better understand why, the distributions of each variable was compared and subsequently ranked from least to most skewed.

Table 50 | Absolute skewness of each individual climate variable and category

Rank	Category	Variable	Type of Skew	|Skewness|
1	Wind Speeds (W)	V-component of wind (v)	Negative	0.17
2		U-Component of wind (u)	Negative	0.27
3	Spatial Features (S)	Geopotential height (z)	Negative	0.33
4	Atmospheric Properties (A)	Temperature (t)	Negative	0.37
5		Relative humidity (r)	Negative	0.55
6		Ozone mixing ratio (o3)	Positive	0.68

It is observed that the more skewed a variable is, the lower the F1-Score. This is evidenced by the fact that the variables of A were the most skewed as shown in Table 50 and any climatological combinations that incorporated A performed the worst as seen in Table 49. To verify this relationship, the model was rerun with power transformations applied to reduce the skewness and the performances were subsequently compared to before. Selvan (2020) states that one of the most popular transformations is the box-cox transformation. Hence, this transformation was applied to combinations (1) and (4) to see if it would result in non-zero hit rates by reducing skewness.

Table 51 | Comparison of results before and after transformation

Atmospheric Properties (A): Combination (1)			
Before/After Transformation	**Recall (R)**	**Precision (P)**	**F1-Score (F)**
Before (Average Absolute Skewness = 0.53)	0%	0%	0%
After (Average Absolute Skewness = 0.10)	0.52%	34.67%	0.52%
Atmospheric Properties (A) + Wind Speeds (W): Combination (4)			
Before/After Transformation	**Recall (R)**	**Precision (P)**	**F1-Score (F)**
Before (Average Absolute Skewness = 0.41)	0%	0%	0%
After (Average Absolute Skewness = 0.06)	0.26%	68.42%	0.52%

Table 51 indicates reducing the skewness does improve the model's performance as evidenced by the non-zero hit rates on high rainfall pixels. Hence, the skewness of climatological variables is another factor that hindered the model's performance with certain climatological variables. These results were further validated by qualitative analysis. For example, in Figure 42, the predicted image after reducing skewness better depicted the top right cluster in comparison to before. Nevertheless, since the goal of this investigation was to identify which variables would result in the best prediction, it is evident from Table 49 that these variables pertain to S and/or W – the geopotential height (z) and wind speeds (u, v) as they are the least skewed.

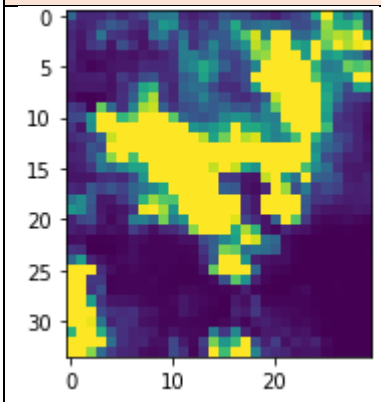
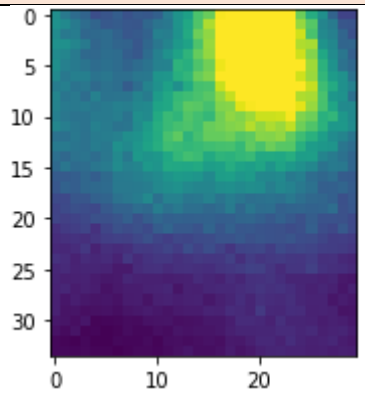
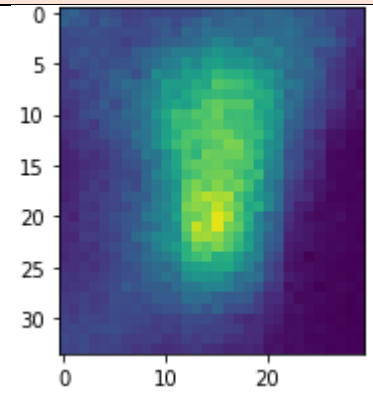
TRUE	PREDICTED (AFTER)	PREDICTED (BEFORE)
		

Figure 42 | Predicted images for combination (1) before and after reducing skewness

5. Comparisons Between CNN_1 and CNN_3

Table 52 | Comparison between CNN_3 (setup 1) and CNN_1 for the same $t_{in} = 3$

Model	t_{in}	Window Size	Recall (R)	Precision (P)	F1-Score (F)
CNN_1	3	5-minutes	21.50%	61.68%	31.89%
CNN_3 (No overlap)	3	5-minutes	0.64%	40.51%	1.26%

The results from Table 52 show that CNN_1 outperformed CNN_3 significantly despite training both models on the same t_{in} and window-size. There are 2 possible reasons for this in addition to the high proportion of zero intensity pixels in the rainfall images, which hampered both models' performance. Firstly, the number of samples for CNN_3 was 22,580 in comparison to the 28,279 of CNN_1. Hence, CNN_3 had fewer opportunities to better understand and map the relationship between input and output more accurately. Secondly, as mentioned in 4.3.2, inaccuracies may have been introduced by duplicating the climate data 10 times every hour to 'artificially' match the temporal resolution of the rainfall images. Of the 22,580 samples in the climate dataset, there are only 2258 unique values for every variable. Hence, duplicating the climate data in such quantities every hour implies that the climatological features are stationary over several 5-minute intervals, thereby introducing inaccuracies. Hence, it would be expected that the model's performance is inversely proportional to the frequency of climate data duplications.

This relationship was validated by rerunning CNN_3 but with $t_{in} = 2$, 5-minute windows and no overlap such that the climate data is now duplicated 11 times for every hour. It can be seen from Table 53 that this setup performed worse with no hits on high rainfall pixels thus verifying the aforementioned explanation. From a qualitative standpoint, CNN_1 was also a better predictor of the general shape and structure of the rainfall field in comparison to CNN_3. This was evidenced by the more obscured predicted image in comparison to the spatial structure captured by CNN_1 as shown in Figure 43, which also verifies the quantitative implications that CNN_1 is a better predictor of high rainfall pixels than CNN_3 for the same t_{in} and window size.

Table 53 | Results from $t_{in} = 2$ with 5-minute windows for CNN_3 (no overlap)

Model	t_{in}	Window Size	Recall (R)	Precision (P)	F1-Score (F)
CNN_3 (No Overlap)	2	5-minutes	0%	0%	0%

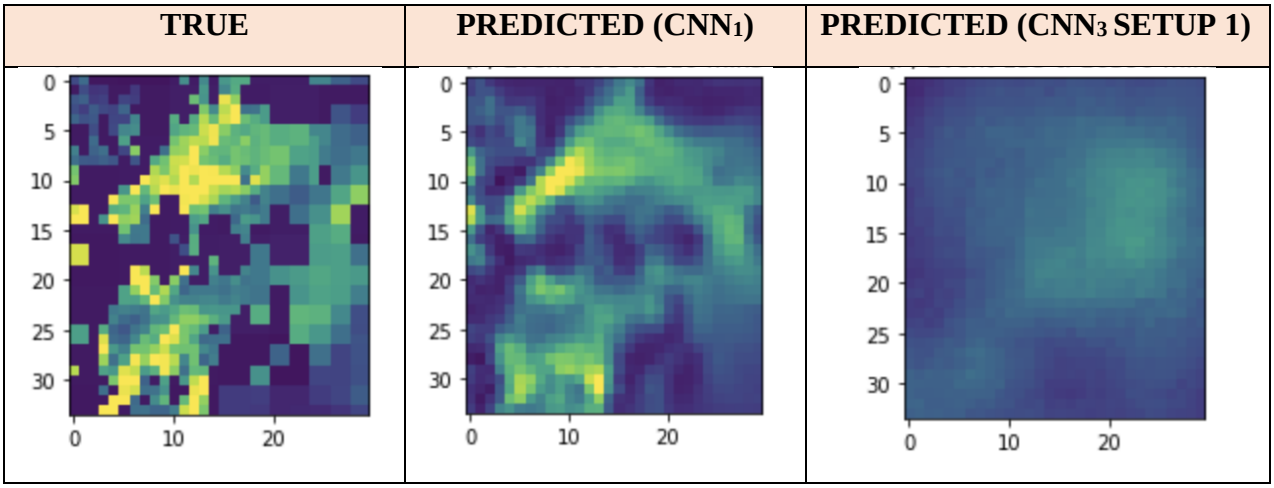


Figure 43 | Comparison of predicted images for CNN_1 and CNN_3

Similarly, with spatially aggregated pixels, both qualitative and quantitative analysis shows that CNN_1 outperformed CNN_3 significantly. This is evidenced by the strong resemblance between the true and predicted image of CNN_1 and the lack of bright pixels in the predicted image of CNN_3 as shown in Figure 44. Where CNN_1's performance improved from reducing the proportion of zero intensity pixels through spatial aggregations, the same cannot be said for CNN_3. This is because changes had to be made to the architecture of CNN_3 to accommodate for spatially aggregated pixels, which subsequently hampered its performance as reasoned in 4.3.4. Nevertheless, the lack of bright pixels in the predicted image of CNN_3 complements the quantitative results that no high rainfall pixels could be predicted by CNN_3 when spatially aggregated.

Table 54 | Comparison between CNN_1 and CNN_3 for spatially aggregated pixels

Model	Recall (R)	Precision (P)	F1-Score (F)
CNN_1 (5km x 5km)	51.06%	78.14%	61.76%
CNN_3 (5km x 5km)	0%	0%	0%

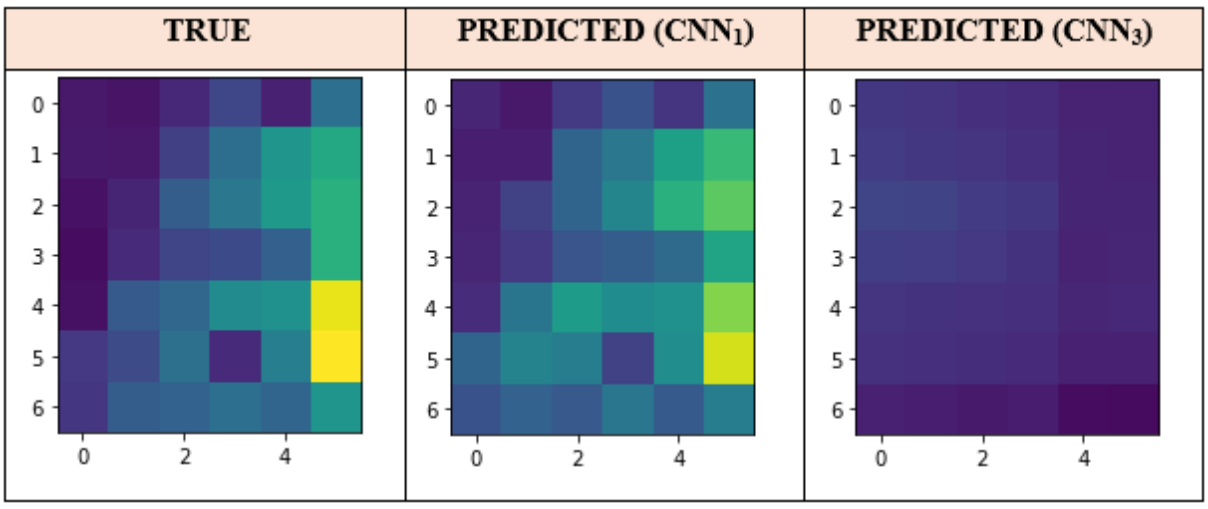


Figure 44 | Comparison of true and predicted images for 5x5 spatially aggregated pixels

Overall, CNN_1 outperformed CNN_3 in every quantitative measure. Both models suffered from the high concentration of zero intensity pixels in the rainfall images, which led to the low recalls observed for CNN_1. However, CNN_3 suffered from this in addition to the mismatching temporal resolutions between the climate data and rainfall images. This necessitated the duplication of climate data, which arguably hampered its performance further by introducing inaccuracies as shown previously. Furthermore, the climatological features had some skewed variables that also adversely affected CNN_3's performance. Qualitatively, CNN_1 was generally better than CNN_3 at capturing the spatial structure of the rainfall field. Furthermore, CNN_3 tended to overpredict the spatial extents of high rainfall clusters in comparison to CNN_1, as evidenced in Figure 41. From an operational standpoint, this is not necessarily a negative feature as it implies that the model is more conservative with the spatial coverage of high rainfall areas. However, both models struggled when smaller and more localised clusters of high rainfall pixels were present and typically depicted a 'smoother' image. This observation also means that the intensities of high rainfall pixels might be underestimated.

From these results, one might be inclined to train a model on rainfall images only (i.e., CNN_1). However, such conclusions cannot be reliably drawn for two reasons. Firstly, the origins of CNN_1 and CNN_3 are fundamentally different. Where CNN_1 was adapted from an investigation on radar-based precipitation nowcasting, which bears strong resemblance to the objectives of this project, CNN_3 was adapted from a network used to predict future frames of a video game. Therefore, the assimilation of climatological features with rainfall images may have been flawed because what worked for frame predictions in a video game may not have worked for rainfall forecasting. Thus, the potential for improvement with CNN_3 is extensive as it has not been adapted for rainfall forecasting purposes before until now (i.e., novel).

Secondly, some unrealistic assumptions were adopted to incorporate the climate data into the model due to the unequal temporal resolutions between the rainfall and climate data, as previously mentioned. Thus, it could be argued that the results are inconclusive since these assumptions do not accurately reflect the temporal behaviour of the climatological features and hence, its effects on the model's predictions have not been reliably simulated. For these reasons, the incorporation of climatological data should not be discounted because this area of research is relatively new and thus, there is potential for further development.

6. Conclusion

In conclusion, CNN_1 could correctly identify high rainfall pixels with a hit rate/recall of approximately 20%. Despite the low hit rates, qualitatively, CNN_1 predicted the spatial structure and extent of the rainfall fields relatively well. After exploring different input formats (increased forecast lead time, missing timesteps, etc.), the hit rate of CNN_1 decreased, indicating that further work is needed to ensure the model's robustness under constrained circumstances. On the other hand, CNN_3, which was trained on both rainfall and climate data, exhibited a significantly lower hit rate of < 1%. Qualitatively, whilst CNN_3 did not predict the spatial structure as well as CNN_1, it was still able to identify clusters of high rainfall pixels and its positions relatively well.

The limitations of this research that resulted in low hit rates lie primarily in the characteristics of the datasets. With both models, the overarching issue was the high proportion of zero intensity pixels in the rainfall images, which enhanced the models forecasts on low intensity pixels in comparison to high intensity ones. However, with CNN_3, the model also suffered from the inaccuracies introduced by duplicating the climate data as exemplified in section 5 but was necessitated by the mismatching temporal resolutions between the rainfall and climate data. Additionally, the skewness in some climatological features from 4.3.5 further hampered CNN_3's performance. It is mainly for these two reasons that CNN_3 underperformed in comparison to CNN_1. Nevertheless, there were also some technical limitations. Namely, the loss function of both models did not minimise the loss on the high rainfall pixels only but rather, attempted to minimise the loss on all pixels. Furthermore, the computational times of both models on average required between 300 – 500s for 1 epoch. Hence, the total number of epochs was limited to 10 in most cases whereas it is general ML practice to train over hundreds of epochs.

These limitations therefore present an abundance of possibilities for future works. For both CNN_1 and CNN_3, 'selected' rainfall images rather than all rainfall images from each event can be used instead to reduce the proportion of zero intensity pixels. This would involve identifying defined periods of high rainfall from each event. Although this would reduce the number of training samples, the number of events incorporated should therefore also be increased to offset this reduction. For CNN_3 specifically, the temporal resolution of the climate data can be refined to equal that of the rainfall data. This ensures that the true climatological data points are associated with the corresponding rainfall images rather than relying on artificially duplicated ones. In both models, the loss function can also be revised to minimise the loss on high rainfall pixels only as opposed to all rainfall pixels.

With these improvements both models may eventually be fit for operational purposes. However, more work is needed to develop such models that are reliable and robust for predicting high rainfall intensities. Although this research has not produced a model that is operationally fit, it has provided beneficial insights to guide future research on the development of rainfall forecasting models.

7. Acknowledgements

I would like to thank my supervisor, Dr Christian Onof for his guidance, encouragement, and for putting his trust in me throughout this research. I would also like to thank Yuting Chen for her patience, unreserved devotion of her time, and willingness to obtain the necessary datasets for me. Both of their technical expertise and insights were invaluable in helping me conduct the necessary investigations. All the outcomes in this study would not have been possible without either of their support.

8. References

Abraham, J. (2017) *Global Warming is increasing Rainfall Rates.* Available from: https://www.theguardian.com/environment/climate-consensus-97-per-cent/2017/mar/22/global-warming-is-increasing-rainfall-rates#:~:text=In%20fact%2C%20relations%20reveal%20that,the%20relationship%20is%20even%20stronger. [Accessed: 11th March 2021]

Ahmed, A. N., Aziz, A., El-Shafie, A., Kushiar, K. F., Ridwan, W. M. & Sapitang, M. (2020) *Rainfall Forecasting Model using Machine Learning Methods: Case Study Terengganu, Malaysia.* Available from: https://www.sciencedirect.com/science/article/pii/S2090447920302069 [Accessed: 17th March 2021]

Anand, A. (2020) *A Tour of the Most Popular Machine Learning Algorithms.* Available from: https://towardsdatascience.com/a-tour-of-the-most-popular-machine-learning-algorithms-b57d50c2eb51 [Accessed: 16th March 2021]

Ayzel, G., Heistermann, M., Lukyanova, O., Nikitin, O. & Sorokin, A. (2019) *All Convolutional Neural Networks for Radar-Based Precipitation Nowcasting.* Available from: https://www.sciencedirect.com/science/article/pii/S1877050919303801 [Accessed: 17th March 2021]

Benshetler, J. (2016) *Why does the Accuracy decrease after I did Data Normalization in a CNN Model?* Available from: https://www.quora.com/Why-does-the-accuracy-decreases-after-I-did-data-normalization-in-CNN-model [Accessed: 24th May 2021]

Bhandari, A. (2020) *Feature Scaling for Machine Learning: Understanding the Difference between Normalization vs. Standardization.* Available from: https://www.analyticsvidhya.com/blog/2020/04/feature-scaling-machine-learning-normalization-standardization/ [Accessed: 17th March 2021]

Bhattacharya, A. (2020) *Effective Approaches for Time Series Anomaly Detection.* Available from: https://towardsdatascience.com/effective-approaches-for-time-series-anomaly-detection-9485b40077f1 [Accessed: 17th March 2021]

Birmingham City Council & Atkins (2012) *Level 1 Strategic Flood Risk Assessment*

Birmingham City Council & WSP (2017) *June 2016 Flooding & Flood and Water Management Act; Section 19 Investigation*

Brownlee, J. (2017) *What is the Difference Between Test and Validation Datasets?* Available from: https://machinelearningmastery.com/difference-test-validation-datasets/ [Accessed: 23rd May 2021]

Brownlee, J. (2019a) *A Gentle Introduction to Pooling Layers for Convolutional Neural Networks.* Available from: https://machinelearningmastery.com/pooling-layers-for-convolutional-neural-networks/ [Accessed: 14th March 2021]

Brownlee, J. (2019b) *Understand the Impact of Learning Rate on Neural Network Performance.* Available from: https://machinelearningmastery.com/understand-the-dynamics-of-learning-rate-on-deep-learning-neural-networks/ [Accessed: 16th March 2021]

Brownlee, J. (2019c) *How to Manually Scale Image Pixel Data for Deep Learning.* Available from: https://machinelearningmastery.com/how-to-manually-scale-image-pixel-data-for-deep-learning/ [Accessed: 23rd May 2021]

Brownlee, J. (2019d) *Impact of Dataset Size on Deep Learning Model Skill and Performance Estimates.* Available from: https://machinelearningmastery.com/impact-of-dataset-size-on-deep-learning-model-skill-and-performance-estimates/ [Accessed: 24th May 2021]

Brownlee, J. (2019e) *How to use Data Scaling to improve Deep Learning Model Stability and Performance.* Available from: https://machinelearningmastery.com/how-to-improve-neural-network-stability-and-modeling-performance-with-data-scaling/ [Accessed: 31st May 2021]

Brownlee, J. (2020) *How to Choose Loss Functions when Training Deep Learning Neural Networks*. Available from: https://machinelearningmastery.com/how-to-choose-loss-functions-when-training-deep-learning-neural-networks/ [Accessed: 16th March 2021]

Bushaev, V. (2017) *Stochastic Gradient Descent with Momentum.* Available from: https://towardsdatascience.com/stochastic-gradient-descent-with-momentum-a84097641a5d [Accessed: 16th March 2021]

Chen, Y. (2021) Email sent to Yen Hann Yoo, 25th February 2021

ClimaTemps (2017) *Rainfall/Precipitation in Birmingham, England, UK.* Available from: http://www.birmingham.climatemps.com/precipitation.php [Accessed: 3rd June 2021]

Danka, T. (2020) *Why are Neural Networks so Powerful?* Available from: https://towardsdatascience.com/why-are-neural-networks-so-powerful-bc308906696c [Accessed: 16th March 2021]

DelSole, M. (2018) *What is One Hot Encoding and How to Do it.* Available from: https://medium.com/@michaeldelsole/what-is-one-hot-encoding-and-how-to-do-it-f0ae272f1179 [Accessed: 29th May 2021]

Durán, J. (2019) *Everything You Need to Know about Gradient Descent applied to Neural Networks.* Available from: https://medium.com/yottabytes/everything-you-need-to-know-about-gradient-descent-applied-to-neural-networks-d70f85e0cc14 [Accessed: 16th March 2021]

Doshi, S. (2019) *Various Optimization Algorithms for Training Neural Network.* Available from: https://towardsdatascience.com/optimizers-for-training-neural-network-59450d71caf6#:~:text=Optimizers%20are%20algorithms%20or%20methods,order%20to%20reduce%20the%20losses.&text=Optimization%20algorithms%20or%20strategies%20are,the%20most%20accurate%20results%20possible. [Accessed: 16th March 2021]

Ferdinand, N. (2020) *A Simple Guide to Convolutional Neural Networks*. Available from: https://towardsdatascience.com/a-simple-guide-to-convolutional-neural-networks-751789e7bd88 [Accessed: 16th March 2021]

Garbade, M. (2018) *Regression versus Classification Machine Learning: What's the Difference.* Available from: https://medium.com/quick-code/regression-versus-classification-machine-learning-whats-the-difference-345c56dd15f7 [Accessed: 17th March 2021]

Ganesh, S. (2020) *What's the Role of Weights and Bias in a Neural Network.* Available from: https://towardsdatascience.com/whats-the-role-of-weights-and-bias-in-a-neural-network-4cf7e9888a0f [Accessed: 11th March 2021]

Gilon, Y. (2021) *CS231n Convolutional Neural Networks for Visual Recognition.* Available from: https://cs231n.github.io/neural-networks-3/ [Accessed: 2nd June 2021]

Grover, P. (2018) *5 Regression Loss Functions All Machine Learners Should Know*. Available from: https://heartbeat.fritz.ai/5-regression-loss-functions-all-machine-learners-should-know-4fb140e9d4b0 [Accessed: 5th June 2018]

IBM (2020) *What are Neural Networks.* Available from: https://www.ibm.com/cloud/learn/neural-networks [Accessed: 11th March 2021]

Irwin, D. (2019) *Severe Flooding set to Increase – Solihull report warns.* Available from: https://www.birminghammail.co.uk/news/midlands-news/severe-flooding-set-increase-solihull-15748088 [Accessed: 3rd June 2021]

Jaadi, Z. (2019) *A Step-by-Step Explanation of Principal Component Analysis.* Available from: https://builtin.com/data-science/step-step-explanation-principal-component-analysis [Accessed: 17th March 2021]

Jackson, A. (2014) *Flooding.* Available from: https://geographyas.info/rivers/flooding/#:~:text=The%20most%20common%20cause%20of,leading%20to%20increased%20surface%20runoff.&text=This%20leads%20to%20a%20sudden,result%20in%20a%20flash%20flood. [Accessed: 11th March 2021]

Keeling, S. (2005) *Is Brum Britain's Tornado Alley?* Available from: http://news.bbc.co.uk/1/hi/england/west_midlands/4339510.stm [Accessed: 3rd June 2021]

Kinha, Y. (2020) *An Easy Guide to Choose the Right Machine Learning Algorithm.* Available from: https://www.kdnuggets.com/2020/05/guide-choose-right-machine-learning-algorithm.html [Accessed: 11th March 2021]

Kostadinov, S. (2019) *Understanding Backpropagation Algorithm.* Available from: https://towardsdatascience.com/understanding-backpropagation-algorithm-7bb3aa2f95fd [Accessed: 16th March 2021]

Kosson, A., Mu, T. & Wang, E. (2017) *Deep Action Conditional Neural Network for Frame Prediction in Atari Games*. Available from: http://cs231n.stanford.edu/reports/2017/pdfs/602.pdf [Accessed: 25th May 2021]

Liu, L., Liu, Y., Ma, X., Yao, W., Yao, Y. & Zhao, Q. (2019) *Short-Term Rainfall Forecast Model Based on the Improved BP-NN Algorithm.* Available from: https://www.nature.com/articles/s41598-019-56452-5.pdf [Accessed: 17th March 2021]

Lopez, R. and Quesada, A. (2020) *3 Methods to Treat Outliers in Machine Learning.* Available from: https://www.neuraldesigner.com/blog/3_methods_to_deal_with_outliers [Accessed: 17th March 2021]

Malik, F. (2018) *Understanding Auto Regressive Moving Average Model – ARIMA.* Available from: https://medium.com/fintechexplained/understanding-auto-regressive-model-arima-4bd463b7a1bb#:~:text=ARIMA%20uses%20a%20number%20of,ARIMA%20relies%20on%20AutoRegression. [Accessed: 11th March 2021]

Malik, F. (2019) *Neural Networks Bias and Weights.* Available from: https://medium.com/fintechexplained/neural-networks-bias-and-weights-10b53e6285da [Accessed: 12th March 2021]

Massam, A. (2020) *The Physics of Precipitation in a Warming Climate.* Available from: https://www.jbarisk.com/news-blogs/the-physics-of-precipitation-in-a-warming-climate/#:~:text=According%20to%20the%20Clausius%2DClapeyron,storm%20systems%20and%20subsequent%20rainfall. [Accessed: 23rd May 2021]

NOAA (2019) *Water Cycle.* Available from: https://www.noaa.gov/education/resource-collections/freshwater/water-cycle#:~:text=The%20water%20cycle%20shows%20the,form%20of%20rain%20and%20snow. [Accessed: 11th March 2021]

NOAA (2021) *Numerical Weather Prediction.* Available from: https://www.ncdc.noaa.gov/data-access/model-data/model-datasets/numerical-weather-prediction [Accessed: 11th March 2021]

NASA (2021) *Global Climate Change Vital Signs of the Planet.* Available from: https://climate.nasa.gov/ [Accessed: 11th March 2021]

Ng, A. (2017a) *C4W1L02 Edge Detection Examples*. [Video] Available from: https://www.youtube.com/watch?v=XuD4C8vJzEQ&ab_channel=DeepLearningAI [Accessed: 12th March 2021]

Ng, A. (2017b) *C4W1L03 More Edge Detection*. [Video] Available from: https://www.youtube.com/watch?v=am36dePheDc&list=PLkDaE6sCZn6Gl29AoE31iwdVwSG-KnDzF&index=3&ab_channel=DeepLearningAI [Accessed: 12th March 2021]

Ng, A. (2017c) *C4W1L06 Convolutions Over Volumes.* [Video] Available from: https://www.youtube.com/watch?v=KTB_OFoAQcc&list=PLkDaE6sCZn6Gl29AoE31iwdVwSG-KnDzF&index=6&ab_channel=DeepLearningAI [Accessed: 12th March 2021]

Ng, A. (2017d) *C4W1L07 One Layer of a Convolutional Net.* [Video] Available from: https://www.youtube.com/watch?v=jPOAS7uCODQ&list=PLkDaE6sCZn6Gl29AoE31iwdVwSG-KnDzF&index=7&ab_channel=DeepLearningAI [Accessed: 12th March 2021]

Ng, A. (2017e) *C4W1L09 Pooling Layers.* [Video] Available from: https://www.youtube.com/watch?v=8oOgPUO-TBY&list=PLkDaE6sCZn6Gl29AoE31iwdVwSG-KnDzF&index=9&ab_channel=DeepLearningAI [Accessed: 12th March 2021]

Perera, A. (2018) *What is Padding in Convolutional Neural Network's (CNN's).* Available from: https://ayeshmanthaperera.medium.com/what-is-padding-in-cnns-71b21fb0dd7 [Accessed: 13th March 2021]

Ruder, S. (2016) *An Overview of Gradient Descent Optimization Algorithms.* Available from: https://ruder.io/optimizing-gradient-descent/index.html#momentum [Accessed: 16th March 2021]

Saha, S. (2018) *A Comprehensive Guide to Convolutional Neural Networks – the ELI5 way*. Available from: https://towardsdatascience.com/a-comprehensive-guide-to-convolutional-neural-networks-the-eli5-way-3bd2b1164a53 [Accessed: 12th March 2021]

Sahay, M. (2020) *Neural Networks and the Universal Approximation Theorem.* Available from: https://towardsdatascience.com/neural-networks-and-the-universal-approximation-theorem-8a389a33d30a [Accessed: 17th March 2021]

Selvan, T. (2020) *Types of Transformations for Better Normal Distribution.* Available from: https://towardsdatascience.com/types-of-transformations-for-better-normal-distribution-61c22668d3b9 [Accessed: 1st June 2021]

Solai, P. (2018) *Convolutions and Backpropagations.* Available from: https://medium.com/@pavisj/convolutions-and-backpropagations-46026a8f5d2c [Accessed: 16th March 2021]

Soni, D. (2018) *Supervised vs. Unsupervised Learning.* Available from: https://towardsdatascience.com/supervised-vs-unsupervised-learning-14f68e32ea8d [Accessed: 17th March 2021]

Udofia, U. (2018) *Basic Overview of Convolutional Neural Networks (CNN).* Available from: https://medium.com/dataseries/basic-overview-of-convolutional-neural-network-cnn-4fcc7dbb4f17 [Accessed: 14th March 2021]

UK Met Office (2011) *Fact sheet No. 3 – Water in the Atmosphere.* Available from: https://web.archive.org/web/20120114162401/http://www.metoffice.gov.uk/media/pdf/4/1/No._03_-_Water_in_the_Atmosphere.pdf [Accessed: 23rd April 2021]

UK Met Office (2021) *UK Extreme Events – Heavy Rainfall and Floods*. Available from: https://www.metoffice.gov.uk/research/climate/understanding-climate/uk-extreme-events-_heavy-rainfall-and-floods [Accessed: 11th March 2021]

Valkov, V. (2019) *Predicting House Prices with Linear Regression | Machine Learning from Scratch (Part II).*Available from: https://towardsdatascience.com/predicting-house-prices-with-linear-regression-machine-learning-from-scratch-part-ii-47a0238aeac1 [Accessed: 17th March 2021]

Vasudev, R. (2019) *Understanding and Calculating the number of Parameters in Convolutional Neural Networks (CNNs).* Available from: https://towardsdatascience.com/understanding-and-calculating-the-number-of-parameters-in-convolution-neural-networks-cnns-fc88790d530d [Accessed: 15th March 2021]

Zhai, Y. (2005) *Time Series Forecasting Competition Among Three Sophisticated Paradigms.* Available from: https://libres.uncg.edu/ir/uncw/f/zhai2005-2.pdf [Accessed: 11th March 2021]

Zhou, V. (2019) *Training a Convolutional Neural Network from Scratch.* Available from: https://towardsdatascience.com/training-a-convolutional-neural-network-from-scratch-2235c2a25754 [Accessed: 16th March 2021]

Appendix A: CNN Layers

Convolutional Layers

Convolutional layers convolve sub-matrices of the image matrix with single or multiple filters through an element-wise multiplication and summation. Filters (kernels) are specific matrices designed to distinguish a particular feature from the image (e.g., a vertical edge filter, a horizontal edge filter, etc.). By convolving the image with the filter, it enables the layer to extract and distinguish important features from the image whilst storing these important features in a reduced form. The convolution operation is also defined by a stride value, which dictates how many steps to slide the filter along the input or image matrix after each convolution step. To demonstrate this process, the following example by Saha (2018) is considered, where an arbitrary filter $\boldsymbol{F}$ is used.

$$Image\ (\boldsymbol{I}) = \begin{bmatrix} 1 & 1 & 1 & 0 & 0 \\ 0 & 1 & 1 & 1 & 0 \\ 0 & 0 & 1 & 1 & 1 \\ 0 & 0 & 1 & 1 & 0 \\ 0 & 1 & 1 & 0 & 0 \end{bmatrix} \qquad Filter\ (\boldsymbol{F}) = \begin{bmatrix} 1 & 0 & 1 \\ 0 & 1 & 0 \\ 1 & 0 & 1 \end{bmatrix} \qquad Stride\ (s) = 1$$

In effect, the filter is 'slid' along the image matrix followed by an element wise multiplication and subsequent summation (Ng, 2017a).

Step 1

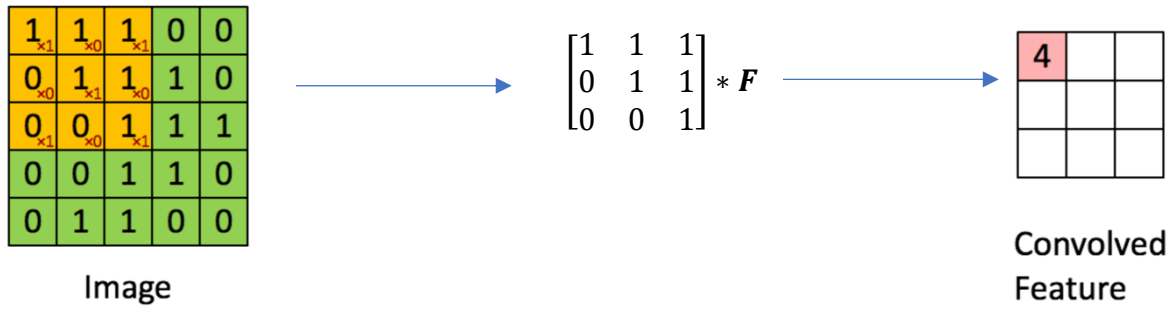


Figure 45 | 1st convolution step (Saha, 2018)

Note: (1x1) + (1x0) + (1x1) + (0x0) + (1x1) + (1x0) + (0x1) + (0x0) + (1x1) = 4

Step 2

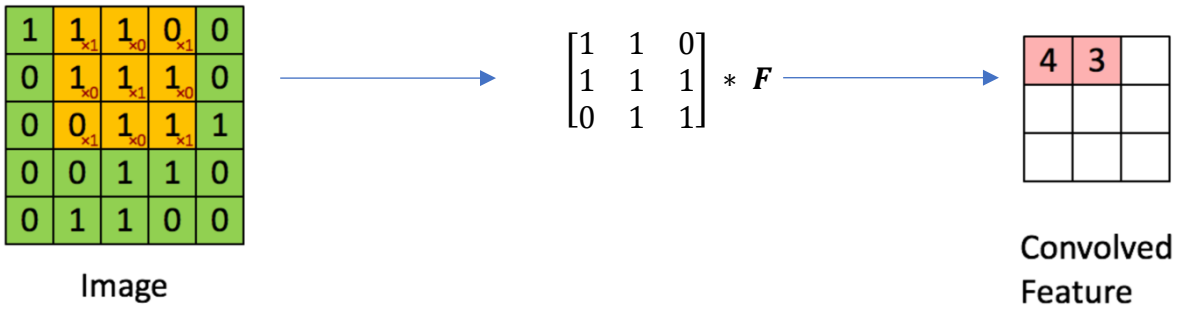


Figure 46 | 2nd convolution step (Saha, 2018)

Note: A stride of 1 means the filter is shifted along by one step from the previous position

Step 3

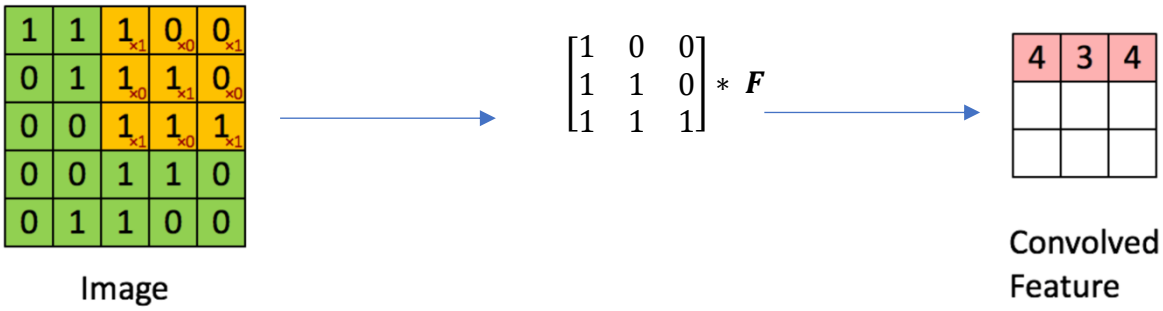


Figure 47 | 3rd convolution step (Saha, 2018)

Step 4

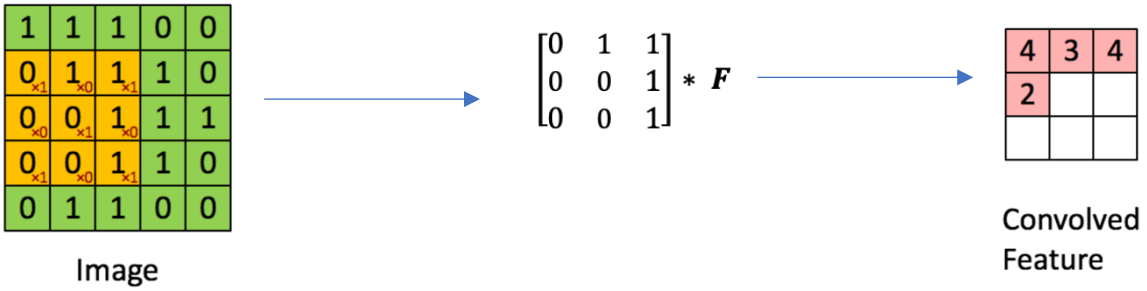


Figure 48 | 4th convolution step (Saha, 2018)

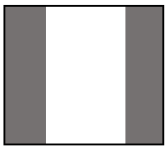
Note: Further steps are not shown here due to redundancy

To demonstrate how convolutional layers and its filters extract important features from an image, a separate example of vertical edge detection in images shown by Ng (2017b) is considered below.

$$Image\ (\boldsymbol{I}) = \begin{bmatrix} 10 & 10 & 10 & 0 & 0 & 0 \\ 10 & 10 & 10 & 0 & 0 & 0 \\ 10 & 10 & 10 & 0 & 0 & 0 \\ 10 & 10 & 10 & 0 & 0 & 0 \\ 10 & 10 & 10 & 0 & 0 & 0 \end{bmatrix} \qquad Filter\ (\boldsymbol{F}) = \begin{bmatrix} 1 & 0 & -1 \\ 1 & 0 & -1 \\ 1 & 0 & -1 \end{bmatrix} \qquad Stride\ (s) = 1$$

In this example, there is a distinct vertical boundary between the 3rd and 4th column of $\boldsymbol{I}$. The filter, $\boldsymbol{F}$, used in this example is known as a vertical edge filter and the intuition here is that the 1's correspond to brighter pixel intensities to the left of the vertical edge while the -1's correspond to darker pixel intensities to the right of the vertical edge, thereby distinguishing a vertical boundary in between. The resulting convolved feature is shown below along with an illustration.

$$Convolved\ feature = \begin{bmatrix} 0 & 30 & 30 & 0 \\ 0 & 30 & 30 & 0 \\ 0 & 30 & 30 & 0 \\ 0 & 30 & 30 & 0 \end{bmatrix} =$$



The vertical boundary feature of $\boldsymbol{I}$ was extracted by $\boldsymbol{F}$ during the convolution process, which is represented by the 30's, with the dimensions reduced from 6x6 to 4x4. Although the example shown is specific to vertical edge detection, the weights/parameters i.e., the elements of $\boldsymbol{F}$ are

usually unknowns and learnable parameters during the training phase of CNNs (Ng, 2017b), such that CNNs will learn what features are important to extract and will determine the weights of the filters accordingly. As such a more generalised representation of a 3x3 $\boldsymbol{F}$ would be given by:

$$\boldsymbol{F} = \begin{bmatrix} w_1 & w_2 & w_3 \\ w_4 & w_5 & w_6 \\ w_7 & w_8 & w_9 \end{bmatrix}$$

As previously noted, CNNs can also handle an additional third/depth dimension known as 'channels.' The most prominent example of 3D inputs are Red, Green and Blue (RGB) images where each channel corresponds to a 2D matrix of a pixel-by-pixel intensity for each colour. Likewise, if the input is 3D, the filters are also 3D with the number of channels in the filter equal to the number of channels in the input.

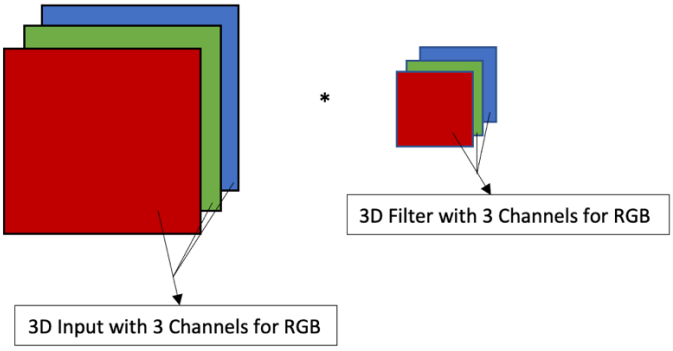


Figure 49 | Illustration of a 3D input and filter

When a convolution operator is applied to a 3D input, each channel undergoes an element wise multiplication with the corresponding filter channel independently of other channels. Subsequently, a summation is carried out to sum the outputs of the element wise multiplication across channels. Saha (2018) provides a step-by-step illustration for the first few steps of this process with a stride of 1, which is shown below.

Step 1

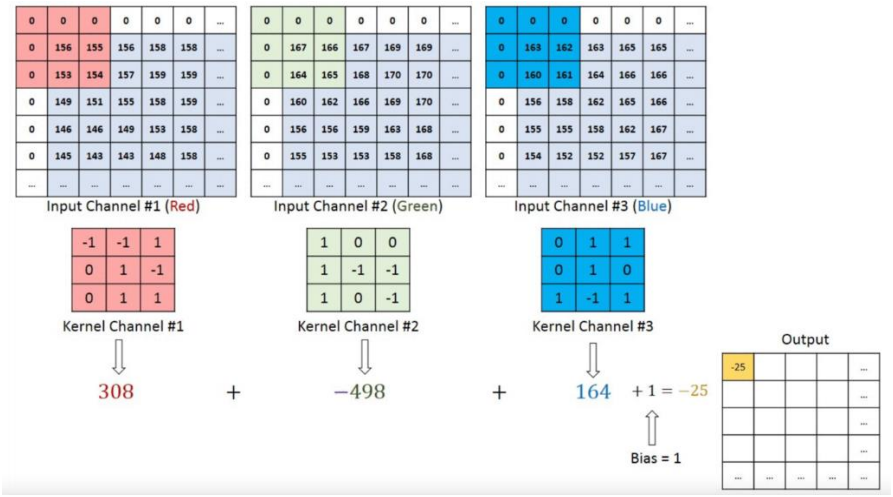


Figure 50 | 1st convolution step (Saha, 2018)

Step 2

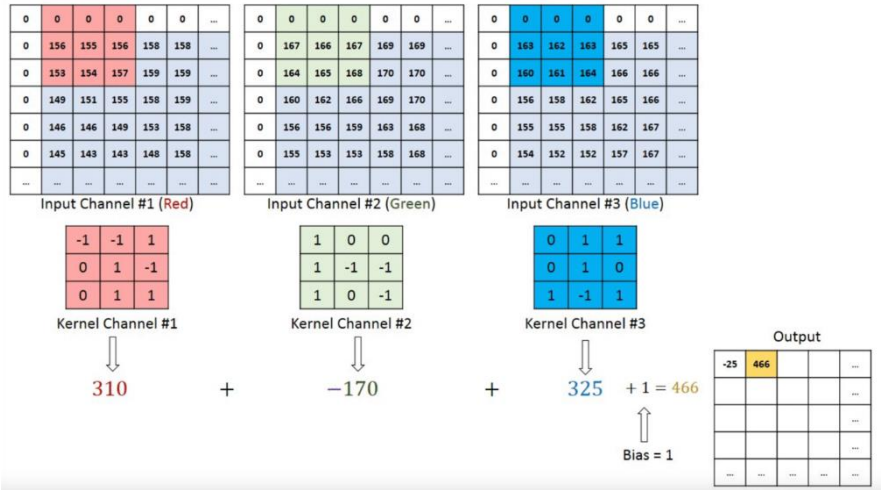


Figure 51 | 2nd convolution step (Saha, 2018)

Note: Further steps are not shown here due to redundancy

It is worth noting that the outputs from the 3D convolution example above is reduced to only one channel. Subsequently, after summing these outputs with bias', they are inputted into an activation function to determine if data is transmitted along that neuron based on whether or not the output of the activation function exceeds a threshold. The most commonly applied activation function in CNNs is the Rectified Linear Unit (ReLU) due to its computational efficiency whereby all negative input values are mapped to zero and hence, these neurons do not get activated. Consequently, only a few neurons are activated each time (Udofia, 2018). Furthermore, ReLU functions do not suffer from saturation in the positive region but do however suffer from saturation in the negative region. This is because all negative inputs [0, -∞] are mapped to the same value equal to zero. The concept of saturation is better illustrated by Figure 52.

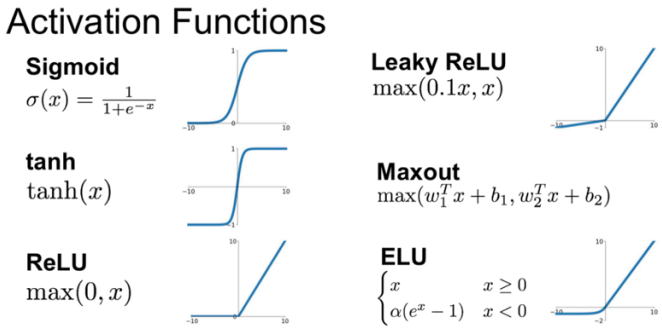


Figure 52 | Different activation functions commonly applied in ANNs (Udofia, 2018)

The outputs from the activation functions are then appended with each other channel wise. It is important to note that the final output of a convolutional layer (the convolved feature) can have fewer or more channels than the filter and input. This is due to the concept of multiple filters. To demonstrate this, Ng (2017c) considers an example of a 6x6x3 input RGB matrix convolved with two separate 3x3x3 filters. These filters therefore correspond to two distinct features the user wishes to detect in the image – vertical edges and horizontal edges. It can be shown that the result from the convolution of the input with each filter yields a 4x4x1 matrix. This example and the corresponding convolutional layer in its entirety are shown in Figure 53 below.

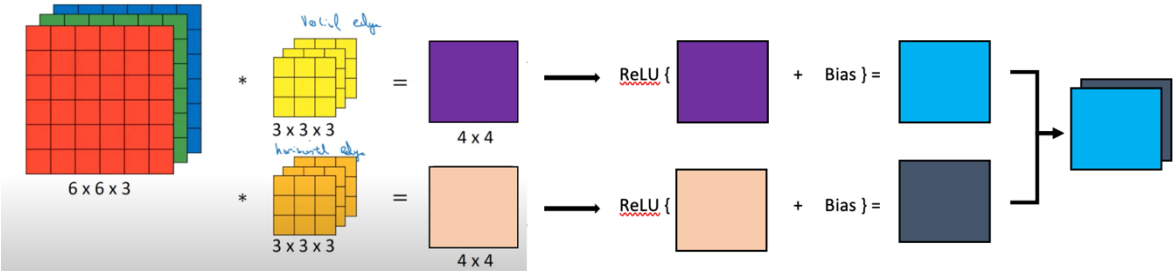


Figure 53 | An example of a full convolutional layer with multiple filters (Ng, 2017d)

Pooling Layers

To motivate the use of pooling layers, it is important to consider the limitations of the convolutional layers. Namely, a key limitation of convolutional layers is that the output records the exact position of the image features. Consequently, small distortions or movements of the features will result in a different output even though the opposite is desired (Brownlee, 2019a). These distortions or movements commonly occur in the data augmentation stage whereby images are stretched or rotated to increase the number of samples in the training dataset. Additionally, larger images can become computationally expensive to convolute due to the increased number of pixels and therefore increased number of features to extract.

A solution to this is down sampling whereby a reduced form of the input is created whilst maintaining the important features without the fine detail. This is commonly achieved through pooling layers. Pooling layers are typically applied after convolutional layers to reduce the spatial size of the convolved feature and therefore decrease the computational power required to process the data – dimensionality reduction (Saha, 2018). In contrast to convolutional layers, the filters in pooling layers do not have learnable weights and biases. Instead, these filters are merely 'sliding windows' over the convolved feature. With these filters, either the maximum value (Max Pooling) or average of all values in that region is extracted (Average Pooling).

The intuition underlying Max Pooling is due to a high number indicating the presence of a feature. Taking the maximum of each region enables preservation of that information in a reduced form. Conversely, if the feature is non-existent or weak in a region, this is reflected by a small number (Ng, 2017e). To demonstrate this, the following example of Max Pooling by Saha (2018) is considered with a 5x5x1 convolved feature, 3x3x1 filter and a stride value of 1.

Step 1

Figure 54 | 1st pooling step (Saha, 2018)

Step 2

Figure 55 | 2nd pooling step (Saha, 2018)

Note: Further steps are not shown here due to redundancy

For 3D convolved features, pooling is applied to each channel independently and the result from the pooling on each channel is then appended channel wise. Hence, the number of channels remains unchanged whilst the height and width are reduced in a pooling layer. It should be noted that Max Pooling acts as a noise suppressant by discarding 'noisy' activations altogether by only taking the dominant feature in each region whilst performing dimensionality reduction (Saha, 2018). On the other hand, Average Pooling performs dimensionality reduction as a noise suppressing mechanism by taking the average of all values in each region. Hence, Max Pooling performs better and is more commonly applied than Average Pooling (Saha, 2018). A comparison of pooling methods is shown in Figure 56 below.

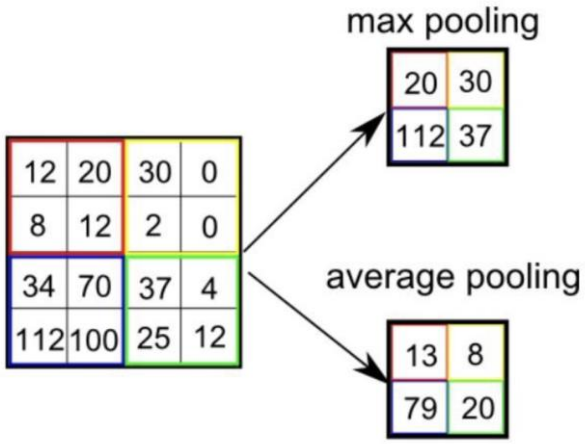


Figure 56 | Comparison of Max and Average Pooling for a stride of 2 (Saha, 2018)

Fully Connected Layers

Fully connected layers function similarly to FNNs and come after convolutional and pooling layers. After a series of convolutional and pooling layers, the output volume is flattened into a column vector. For example, a 14x14x3 output will be flattened into a 588x1 column vector. In the fully connected layer, this flattened output volume is subsequently fed as an input layer into an FNN, and backpropagation is applied to every iteration of training to update and learn the weights (Saha, 2018). Over a series of epochs (a hyperparameter denoting the number of times the entire training dataset is forward passed through the network from input to output), the model will be able to distinguish features and eventually classify the original image using activation functions such as softmax (Saha, 2018). For example, in Figure 4, a series of

probabilities will be outputted for each vehicle class and the maximum probability will correspond to the vehicle the network believes to be represented in the image.

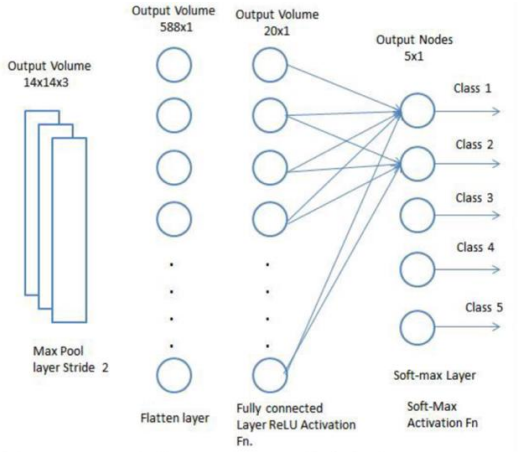


Figure 57 | Example of a Fully Connected (FC) layer (Saha, 2018)

Appendix B: Datasets and Code

The Python codes and datasets are available at:

https://imperiallondon-my.sharepoint.com/:f:/g/personal/yhy17_ic_ac_uk/Eudehyx2ymVArv7llN_MalwBKETh2LBAgoZTMnFNBhL6uw?e=kufNiX