Universiteit Leiden

Master Computer Science

FlaSH: Lightning Prediction Using GOES-16 Satellite Images

Name:	Omid Tajalizadehkhoob
Student ID:	S2896370
Date:	24.03.2023
Specialisation:	Artificial Intelligence
1st supervisor:	Fons Verbeek
2nd supervisor:	Cor Veenman

Master's Thesis in Computer Science

Leiden Institute of Advanced Computer Science (LIACS)
Leiden University
Niels Bohrweg 1
2333 CA Leiden
The Netherlands

Contents

Acknowledgements

First of all, I would like to thank my Family. My mother and father, who supported me without reservation during my whole academic career. I would want to thank my sister, Samaneh Tajalizadehkhoob, for giving me her unbridled support, when I was in need the most.

Secondly, I'd like to express my gratitude to my supervisors, Jos De Laat, Cor Veenman, and Jasper Wijnands. You have given me so much consideration and have never given up on me or my work, even when it seemed impossible at times.

In addition, I would like to thank my master's degree professors for paving the way for me to accomplish my goals. It has been a privilege to be your student and to learn from you.

Abstract

Lightning is a natural weather hazard still claiming thousands of lives globally each year. Prevention of fatalities due to lightning via rules and regulation is common practice in society but comes with costs and delays. Skillful prediction of the occurrence of lightning is therefore an important aspect of weather forecasting. Traditionally, lightning occurrence is predicted by the use of physics-based numerical weather prediction models. However, such model simulations are computationally expensive and tend to not incorporate the latest observational information into the prediction. Here, we follow a different approach by exploring the potential of the use of learning algorithms applied to satellite observations of clouds over the Caribbean region for providing skillful short-term lightning predictions for the following 35 minutes (so-called "nowcasting"). Lightning is associated with a particular type of weather and clouds (thunderstorms). Geostationary satellites nowadays provide continuously updated near-real-time high-quality cloud information at high spatial resolutions. Hence, the possibility exists that satellite observations contain valuable information about potential near-future lightning occurrences.We train well-known models like ResNet50 and MobileNetV3 as well as a basic convolutional neural network (CNN) individually on 100,000 patches with an 8km by 8km resolution, taken from the original satellite images over the entire BES islands (Bonaire, Saint Eustatius, and Saba) region, to achieve 0.99 accuracy and 0.99 AUC score. By performing a convolution-like operation on patches of the entire area satellite picture, we introduce a novel approach called FlaSH (Flash Score Heatmap). For the entire satellite picture, FlaSH achieves an AUC score of 0.96, demonstrating its ability to precisely nowcast lightning.

Chapter 1

Introduction

Lightning is an important and potentially lethal natural hazard claiming thousands of lives globally every year [9]. There exists a range of human and economic activities for which the presence of lightning leads to – sometimes mandatory - changes in activities and operations, and resulting in disruptions and delays. In addition, lightning is associated with several geophysical hazards such as severe convection and explosive and dangerous volcanic eruptions. Knowing when and where lightning occurs and how it changes saves costs and lives.

One of the responsibilities of The Royal Netherlands Meteorological Institute (KNMI) is weather forecasting and natural hazard monitoring for three Dutch Caribbean overseas territories - also known as the BES islands after their names Bonaire, saint Eustatius and Saba. In addition, two of these islands – saint Eustatius and Saba – are of volcanic nature. KNMI has started building a – remote/virtual – weather and volcano monitoring infrastructure. This includes building a satellite data platform for the BES islands, the so-called "BES dashboard".

KNMI has taken a particular interest in predicting the occurrence and changes of lightning activity for the aforementioned reasons. However, numerical approaches for weather prediction of the occurrence of lightning can be rather difficult, especially in the tropics and subtropics, where thunderstorms can develop in a matter of minutes and which are notoriously difficult to predict by numerical weather models. Furthermore, predicting volcanic eruptions is challenging. Little particles of volcanic material collide with one another at high speeds in the cloud of a volcanic eruption. These collisions may lead to the separation of charges in the cloud, resulting in lightning. Hence detection of lightning over a volcano can be an important first indicator of a starting and developing eruption [20].

KNMI has adopted a novel approach by exploring the possibilities of observation-based short-term lightning prediction – the next few hours, also known as "nowcasting". Observations are derived from the American geostationary GOES-16 satellite, which apart from carrying instruments to provide images of earth and in particular of clouds in multiple visible and infrared wavelength bands also carries a lightning detector, the so-called Global Lightning Mapper (GLM).

To make maximum use of the richness of data the approach adopted here is to develop an Artificial Intelligence/Machine Learning algorithm (AI/ML) for GOES cloud information-based probabilistic prediction of lightning several hours ahead in time. Within the field of meteorology, applied AI/ML has yet to be fully integrated and reach maturity. Scientists in this field have long been cautious to apply this technology until recently. Forecasters build mental maps (a model of the environment which is built up over time in the individual's brain) of the actual physical state of the atmosphere based on models, basic physical principles and observations [2, 7, 4]. Forecasters, as well as scientists, have a basic understanding of the limitations of verification statistics and are thus not easily confused by excellent results. Coupled with its novelty and unfamiliarity with AI leads to (some) resistance. For example, KNMI has only started on developing its own AI/ML strategy in 2022. This project therefore also serves as an early example of application development that fit with the KNMI AI/ML strategy.

In this master's thesis research project we compile a robust dataset of satellite observation samples. The data from satellites is delivered in large quantities and a sophisticated structure. Correctly extracting the pertinent information from these files is vital and involves various obstacles. In addition to the huge volume, complex structure and limited tools to deal with these data files, missing data is one of the most significant obstacles when producing samples from such large datasets. We aim to develop a method that can reliably extract patches(feature vectors) from satellite images.

Furthermore, we explore utilizing well-known machine learning models such as ResNet50 [5], MobileNet [10], DenseNet [6], and our implemented SimpleCNN for binary image classification, using the created dataset. Additionally, by convolving the trained model over a full region, we aim to Heatmaps that can accurately show the regions with the highest probability of flash occurrence 35 minutes ahead of time.

1.1 Problem Statement

Lightning forecasting is affected by a variety of atmospheric factors. Existing studies and applications have shown that by examining data from the aforementioned channels, it is feasible to practically forecast the occurrence of lightning [18]. In our research we focus on the chance of lightning happening in the whole region, meaning that by creating patches with spatial resolution of 32km $\times$ 32km from full ABI images and applying binary image classification, we aim to generate an accurate lightning occurrence probability heatmap of the BES island region.

Chapter 2

Related Work

Focusing not only on lightning but on intense thunderstorms [13], this research proposes a deep-learning model to prediction detect intense midlatitude convection formation using geostationary satellite images. The data used in this network consists of 0.64 μm reflectance, 10.35 μm brightness temperature, and flash-extent density data from the GOES-16 satellite's Advanced Baseline Imager(ABI) and Geostationary Lightning Mapper (GLM). Using a training dataset of over 220,000 human-labeled images, The proposed model achieved high performance in detecting intense midlatitude convection (thunderstorms), with an accuracy of 96.9, precision of 96.8, and recall of 96.9. ABI $10.35 - \mu m$ brightness temperature. GLM flash-extent density and $0.64 - \mu m$ reflectance was selected as the most relevant predictors, and the model provides forecasters with quantitative information that often anticipates the onset of severe weather.

Other approaches have accomplished lightning nowcasting by utilizing machine learning methods such as decision trees [11] and support vector machines (SVM) [15]. In more recent studies, semantic segmentation has been highlighted. LightningNet [16] investigates is a novel semantic segmentation deep learning network for cloud-to-ground lightning nowcasting over the next hour. It uses many sources of observation data, including data from a geostationary meteorological satellite (Himawari-8), Doppler weather radar network, and cloud-to-ground lightning localization system, in order to generate more accurate forecasts. The network was trained to extract characteristics of lightning start, development, and dissipation. The proposed approach achieved an accuracy of 0.96 for predicting lightning occurrences with a lead time of 10 minutes, and an accuracy of 0.90 for predicting lightning occurrences with a lead time of 60 minutes.

We have also seen the use of Long Short-Term Memory (LSTM) [12, 14] networks for lightning forecasting. One of the most recent approaches in lightning quantity prediction [19] uses remote sensing data and a CNN-LSTM to predict the amount of lightning in the following hour. This network was able to obtain a Mean Absolute Error (MAE) of 51 (flashes/h^{-1}), indicating that it only incorrectly predicts 51 flash occurrences in the next hour.

Chapter 3

Dataset

3.1 Data Source

The Geostationary Lightning Mapper (GLM) is an instrument on board the Geostationary Operational Environmental Satellite (GOES) series of weather satellites operated by the National Oceanic and Atmospheric Administration (NOAA). GLM is designed to measure total lightning activity in real-time over the Americas and adjacent ocean regions, providing valuable information for weather forecasting and monitoring of severe weather events.

The GLM instrument uses a photodetector to measure the light emitted by lightning flashes, which is then processed by onboard electronics to determine the location, timing, and characteristics of lightning activity. The instrument operates in the near-infrared spectral range, with a peak sensitivity at around 777.4 nm. It has a spatial resolution of approximately 10 km and provides data with a temporal resolution of 20 seconds while allowing for the detection and tracking of individual lightning flashes and the evolution of lightning activity over time.

The GLM instrument provides a range of measurements that can be used for scientific analysis, including the total number of lightning flashes, the flash rate density, the flash extent density, and the flash energy. These measurements can be used to study the dynamics of thunderstorms and the formation of severe weather events, as well as to evaluate the performance of lightning detection algorithms and to validate other lightning measurements from ground-based networks or other satellite instruments.

GOES-16 is the operational geostationary weather satellite that focuses on the Americas. It is located at 75.2 degrees west above the equator at a distance of approximately 36000 km from earth. With its Advanced Baseline Imager (ABI), GOES-16 is able to capture high-resolution images of the Earth in 16 visible and infrared spectral bands, allowing for a wide range of applications [10]. The Geostationary Lightning Mapper (GLM) on GOES-16 was the first lightning mapper to function while in a geostationary orbit. In addition to these two primary sensors, the spacecraft carries four additional pieces of scientific equipment for studying the Sun and space weather.

In this research, we concentrate on three of the 16 wavelength [23] channels that the

Advanced Baseline Imager (ABI) uses to view the earth and the atmosphere. These channels have been analyzed using conventional and complex numerical models for predicting lightning and have already been shown to contain useful information about clouds that grow into thunderstorms, the development of those thunderstorms, and their chances of producing lightning [8]. We picked precisely three channels to generate samples that resemble RGB images, making it simpler to feed them to our models. The selected channels are:

- **Channel 2 (0.64 μm)** is a visible channel with a spatial resolution of 0.5km, As the second of the two channels in the visible spectrum, this one is closer to the red end of the spectrum. Daytime snow and ice cover, severe weather detection, low-level cloud-drift winds, smoke, volcanic ash, hurricane analysis, and winter storm analysis are visible phenomena that can be studied using this channel information.

- **Channel 13 (10.3 μm)** is an infrared channel with a spatial resolution of 2km. Since it is less susceptible to water vapor than other infrared window channels, this one is termed "clean." As a result, it is able to see through thinner layers of cloud and get a clearer glimpse of ice.

- **Channel 15 (12.3 μm)** is an infrared channel with a spatial resolution of 2km. Since this channel is more sensitive to water vapor than others in the infrared window, it is referred to as the "dirty" channel. It is used to calculate the split window difference [22] when compared to the "clean" window (channel 13). Subtracting the numbers at one place on the "dirty" channel from those at the same location on the "clean" channel constitutes a split window difference. Under daylight, this helps to emphasize moisture discrepancies.

The amount of incoming data from the GOES-16 satellite is substantial. Even with chosen days of the year, our first data included more than 19 thousand data files. These data files have a complicated structure, as each file provides data from 16 wavelength channels. We sought to isolate the most important aspects of this data and transform it so that it could be fed into a neural network. To do this, we extracted the radiance values from each data file. We were only interested in a specific region so processing all data was an unnecessary use of resources. Also, it would considerably increase time needed for processing and training. Since our interest was in the BES islands, we used the NCKS library to extract the ABI data for our chosen region (lat, lon of the desired area) from these data files. This library is capable of extracting a region from the ABI dataset as well as all the characteristics corresponding to that region. Our data is from 2019. The KNMI meteorologists carefully selected a set of days to ensure that each day would have some lightning within the 3000–5000 km area and in the vicinity of one of the BES islands. These were days that the KNMI meteorologists responsible for the BES weather forecasting monitored lightning development as part of their regular procedures. This was done with the understanding that even on days with lightning, there is a substantial probability of clear skies and other cloud activity on the same day, so there is no bias towards severe convection or certain types of samples. The data is obtained from the noaa-goes16 S3 data bucket on the Amazon Web Service making use of information provided by [24].

3.2 Challenges

The eventual volume of ABI and GLM data for the selected days in the area is quite large. Since the data files were extracted every 10 minutes, we had 144 data files per channel every day, which yields more than 19,000 data files. The exact days used for this task are mentioned in Appendix A. The first obstacle to overcome was the missing data. There were several forms of missing data :

1. time periods that were entirely absent from all channel files for each 10-minute period.
2. missing certain channel files, implying one or two channel files were missing for those precise 10-minute periods.
3. channel files that were missing some radiance values inside the data files.
4. missing corresponding GLM files to produce labels for the feature vectors.

We were able to solve these challenges by developing a check function that was specific to each unique challenge.

ABI data is provided at a regular longitude-latitude grid, whereas GLM data consists of a list of events each with a time and coordinate. Each flash coordinate had to be assigned to a particular ABI pixel and/or a collection of ABI pixels covering a predefined area. We utilized the cv2 library and interpolation to increase the resolutions of channels 13 and 15 to be the same as the resolution of channel 2.

After this conversion, the dataset was ready to link each GLM file to its matching ABI files.

3.3 Feature Vector Creation

Each sample consists of a feature vector and a target label 1 for Flash samples and 0 for No Flash samples. For our flash data, we utilized data from GLM. Due to the predictive nature of our work, for our feature vectors, we focused on cloud data from ABI, leading GLM data by 35 minutes, meaning that while looking at cloud data for a certain time, we had the data of the flash events of 35 minutes later. From the GLM files, we retrieve the coordinates of the individual flash occurrences and converted them to pixel values before using them to generate our samples. As for our cloud data and as explained in Section 1, we use three bands (i.e., channels) of the ABI imager. We retrieve the radiance measure of each channel and converted it to brightness temperature. We then scale these pixel values to make sure they stay within the $[0, 1]$ bounds. The brightness temperature is a metric for describing the amount of heat radiated by the atmosphere to the satellite through microwaves, as stated in terms of the temperature of an analogous black body.

Since we made use of two different types of channels which are visible and infrared, we needed two different methods to convert the radiances to brightness temperature. For the visible channel 2 we have the following [1]:

$$BT_{ch2} = RAD_{ch2} * \kappa_0 \tag{3.1}$$

With RAD being the radiance measurement, and κ_0 being a conversion factor to convert radiances (W/m^2) into the reflected fraction of sunlight taking geometric effects like solar angles into account. As with the other two infrared channels, channel 13 and channel 15, the conversion has to be done using a different formula [1]:

$$BT_{ch13,ch15} = \frac{\frac{pfk_2}{\log \frac{pfk_1}{RAD} + 1.0} - pbc_1}{pbc_2} \tag{3.2}$$

with pfk and pbc being conversion factors provided in the ABI data files. Beyond these two transformations, we stacked the data from the three channels to generate an RGB format. This format is the default input format for all the models we intend to utilize in this paper. Figure 3.1 illustrates instances of the brightness temperature for each of the three channels.

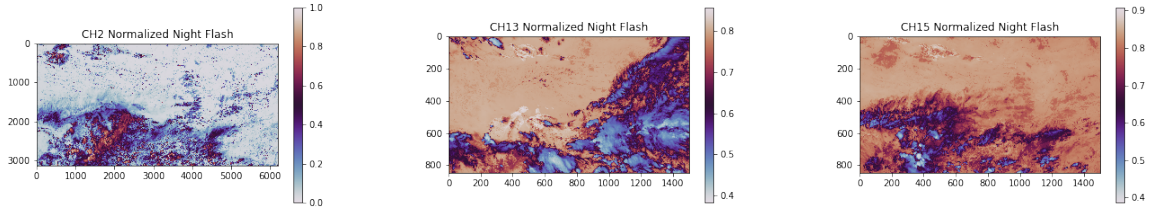

Figure 3.1: Visualized brightness temperature data for the full region for each channel. From left to right we have the data for channel 2, channel 13 and channel 15 respectively. We can see that channel 2 data is approximately 3500×6200 pixels, due to the pixel sizes being $0.5km \times 0.5km$, and in channels 13 and 15 we have approximately 850×1500 pixels due to pixels being $2km \times 2km$. Images from channel 2 and 13 were taken late in the afternoon (22:00 UTC). Channel 13 image was taken at 23:00 UTC.

It is important to note that, channel 2 is a visible channel. Because only the visible section of the light spectrum is used to make a visible satellite picture, its use is limited to daytime hours. Hence our samples include feature vectors with values close to 0 for channel 2. In Figure 3.2 we can see the brightness temperature images taken in the early morning and at night.

Our dataset consists of samples converted to brightness temperature. Due to full BES region area being 3000km $\times$ 1500km, there were several cloud systems present in each picture. To guarantee that our model can discriminate distinct cloud systems, we opted to collect smaller samples (about 8km $\times$ 8km). Feature vector creation and labeling comprised of two phases. The process has been illustrated in Figure 3.3:

- **Positive Samples**. We also refer to them as *Flash* samples. Meaning that a flash event has occurred for this region and these atmospheric conditions during the following 35 minutes. We obtained this data by locating the pixels where the flash events occurred using the GLM data. After identifying these pixels, we extracted a 64×64 pixel region(approximately 32km $\times$ 32km) around them. We ensured that the flashes in our samples occurred at various positions and were not always in the central pixel. Examples of the Flash samples for each individual channel are shown in Figure 3.4.

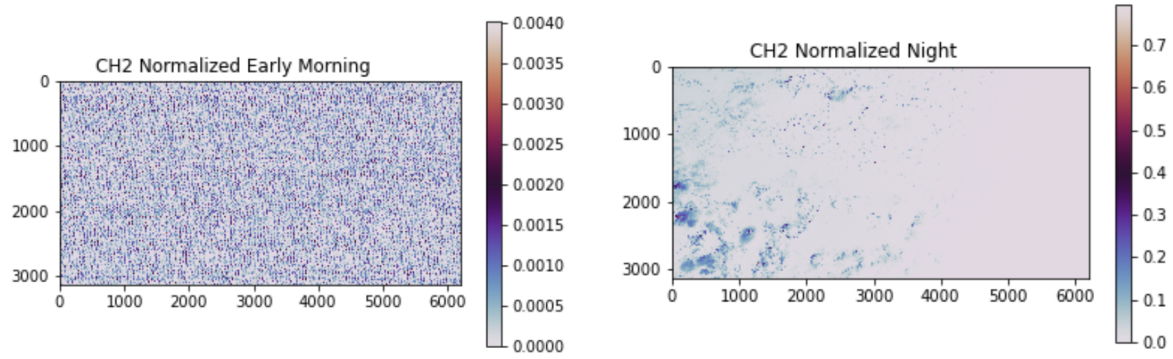

Figure 3.2: Visualized brightness temperature data for channel 2. The image on the left was captured in utter darkness at 3:00 UTC, and we can see that the values reduce towards zero. The image on the right was captured at 20:00 UTC, when there are a few clouds visible due to fading light.

- **Negative Samples**. These are also referred to as the *NoFlash* examples. We wanted the data to be as diverse as possible, meaning that the negative samples are from a variety of scenarios such as clear sky, some visible clouds, cloudy sky with no flashes etc. At each iteration, we chose a random pixel from each picture. By checking all the surrounding pixels, if no flash events were present, we extracted this sample and designated it as our NoFlash sample. Owing to the nature of these samples and the necessity to examine all surrounding pixels for flash events, the production of negative samples required extra computing resources.

	Flash	NoFlash	All
Train	42000	42000	84000
Validation	11554	12075	23629

Table 3.1: Quantity of the samples created to train our classification models. In total, more than 100,000 samples were created to be able to train our network.

We created approximately 100,000 samples with the negative and positive sample datasets being almost identical in set size. Therefore the dataset is balanced and we can rule out all the negative impacts that an imbalanced dataset can have on model training and performance and estimation metrics. Figure 3.1 shows an example of the cloud characteristics. The dataset can be found at Dataset.

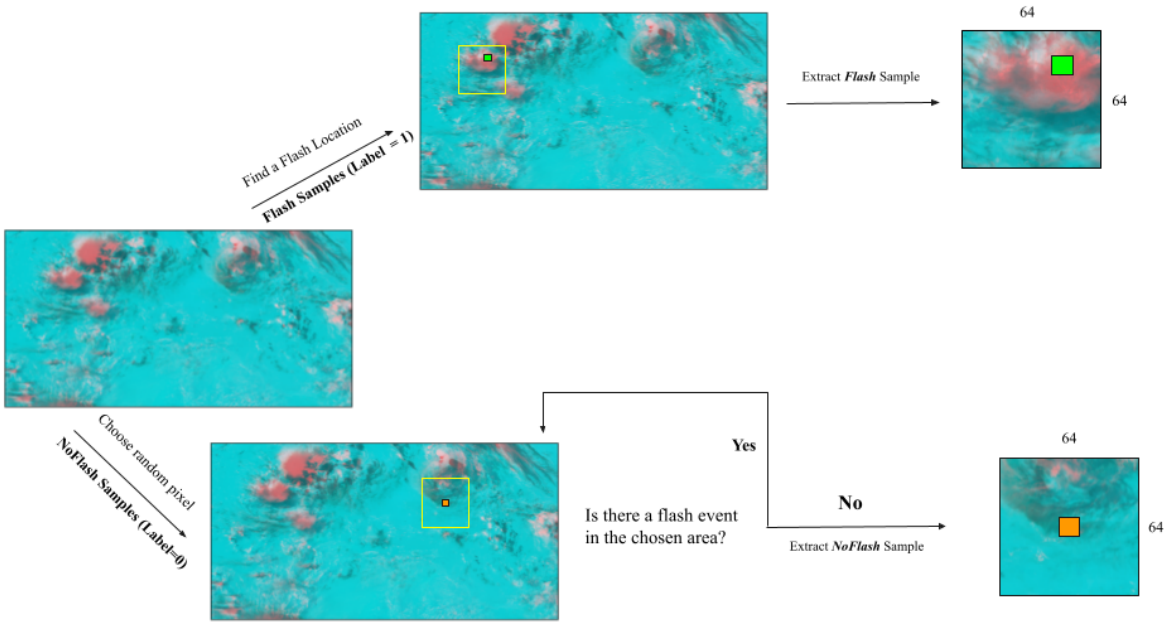

Figure 3.3: A diagram illustrating the sample extraction procedure. At the top process of extracting the Flash(Positive) samples is shown. The procedure for extracting NoFlash (negative) samples is illustrated at the bottom.

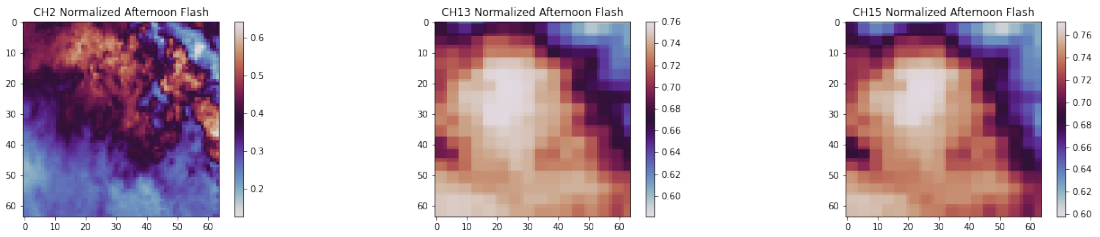

Figure 3.4: 64×64 flash samples in the afternoon time (15:00 UTC) for channels 2,13,15 from the left respectively. As we can see, channel 2 has a higher resolution, and the other two channels, due to the resizing for sample creation, have visible squares in the image.

Chapter 4

Methodology

Our approach consists of two major components: model learning and model application at sample and region level. In the first section of this chapter, we outline our technique and the classification models used to classify the samples we generated. Finally, we present our approach for constructing a heatmap of the whole desired area in order to pinpoint the spots where flashes are most likely to occur. Our ultimate objective is to identify the network that can provide the best accurate heatmap of lightning locations.

4.1 Modelling

In this section, we describe our approach to the model learning problem. Throughout this paper, we utilize binary image classification to classify between lightning and none lightning patches. To achieve this, we utilized three well-known image classification models, ResNet50 [5], MobileNetV3 [10] and DenseNet [6]. These models are known to have high accuracy on image classification problems and are trained on the ImageNet [3] dataset which has 1000 classes. We modify the models to be able to do binary classification. The modifications in all cases were quite similar. We use of the built-in models in the Keras library for MobileNetV3Small, ResNet50, and DenseNet121. A list of the modifications to the default version of Keras networks are:

- Input Shape: We have changed the input shape from the default which is $(224, 224, 3)$ to our feature vector shape which is $(64, 64, 3)$.
- Number of Classes: This parameter determines the number of output classes of the classifier. If the weights of ImageNet are used, this parameter should be set to 1000. In our case, since we have a binary classifier, we want this model to predict a probability in $[0, 1]$. We set this parameter to 1 and change our activation function accordingly.
- Classifier Activation: Since we aim to create a prediction in the range of $[0, 1]$, we chose the sigmoid activation function which is the best suited for binary classification tasks and probabilistic prediction.

4.1.1 ResNet50

ResNet[5], which is short for Residual Networks, is a type of Convolutional Neural Network (CNN) architecture that belongs to the family of "Deep Residual Learning" models. It is a traditional neural network that acts as the foundation for many computer vision applications. Instead of simply approximating the desired underlying mapping, ResNets learn residual functions by making use of the layer inputs, which is the basic concept behind ResNets. The vanishing gradients issue, which may arise in extremely deep networks, is avoided by using residual blocks, allowing for the training of networks much deeper than 100 layers. Since its release, the ResNet architecture has been extensively employed in various computer vision tasks due to its ability to achieve state-of-the-art performance on image recognition tasks such as the ImageNet dataset.

The key part of ResNets are the residual blocks. residual block is a building block that is composed of two or more convolutional layers with batch normalization and ReLU activation functions. Figure 4.1 shows this structure. The formula for a residual block is as follows:

$$y = F(x, W_i) + x \tag{4.1}$$

where x is the input to the block, $F(x, W_i)$ is the output of the convolutional layers with weights W_i, and y is the output of the residual block.

The main idea behind the residual block is that it allows the network to learn a residual function, which is the difference between the desired underlying mapping and the output of the convolutional layers. The residual block adds the input x to the output of the convolutional layers, allowing the network to learn the residual function rather than the underlying mapping directly.

4.1.2 MobileNetV3

MobileNetV3 [10] is a convolutional neural network optimized for mobile and embedded devices. Utilizing Mobile Inverted Bottleneck Blocks is one of its important characteristics. Using a mix of 1x1 and 3x3 convolutions coupled with a squeeze-and-excitement method, these blocks are intended to enhance feature representation with fewer parameters. The squeeze-and-excitement process enables the network to learn which properties are most essential and emphasize them more while deemphasizing others that are less significant. The Mobile Inverted Bottleneck Blocks additionally provide a nonlinear bottleneck between the block's input and output, which further enhances feature representation while lowering the number of parameters.

Here are some aspects that make MobileNetV3 unique:

1. **Hard Sigmoid Activation Function.** MobileNetV3 uses a hard sigmoid activation function, which is a variant of the sigmoid function that is more computationally efficient. The hard sigmoid function has a linear region in the middle and zero gradients outside of that region, which makes it more efficient to compute than the

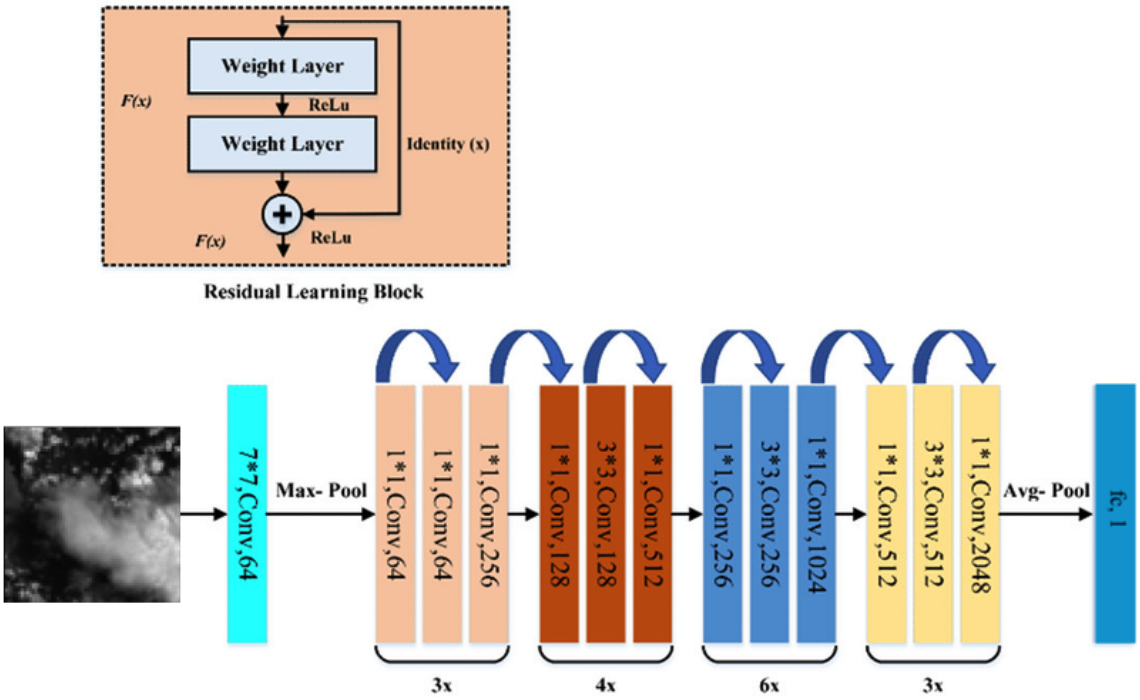

Figure 4.1: ResNet50 architecture [17]. input is a 64 × 64 samples, and the final layer produces a probability.

traditional sigmoid function. This allows the network to achieve high accuracy while using fewer computations. The formula is as follows:

$$X = \begin{cases} 0 & x < -2.5 \\ 1 & x > -2.5 \\ 0.2x + 0.5 & -2.5 \leq x \leq 2.5 \end{cases} \tag{4.2}$$

2. **Combination of Multiple Small Filters and Global Pooling.** MobileNetV3 utilizes a mix of several small filters and global pooling to significantly minimize computation while preserving accuracy. Instead of using larger filters, the network employs a collection of smaller filters, enabling it to capture finer input features while also lowering computation. In addition, the network employs global pooling, which minimizes the number of parameters by substituting a global average pooling layer for the fully connected layer. This reduces computation while preserving precision.

3. **Multiple Size Options.** This network offers many size choices to balance model size and accuracy. This implies the network is adaptable to various computing restrictions and device capabilities. By selecting a lower size, the network may be made more compact and efficient at the expense of some precision. In contrast, selecting a bigger size enables the network to attain more accuracy, but at the expense of a larger model size and increased computational complexity.

4.1.3 DenseNet121

DenseNet121 [6] is a CNN architecture that is commonly used for image classification tasks. This network is an extension of the popular ResNet architecture. The main idea

behind DenseNet is to create dense connections between layers, where each layer is connected to every other layer in a feed-forward fashion. This allows the network to reuse features learned at different layers throughout the network, which leads to better feature reuse and better accuracy.

DenseNet121 consists of 121 layers, including a convolutional layer, four dense blocks, and a classification layer. Each dense block contains a set of convolutional layers, followed by a batch normalization layer and a ReLU activation function. The output of each block is concatenated with the input to the block, forming a dense connection. This allows each layer to have access to the output of all previous layers, facilitating feature reuse.

In each dense block of DenseNet121, each layer is connected to every other layer in a feedforward fashion. The output of each layer is concatenated with the outputs of all previous layers and passed as input to the next layer. This dense connectivity is illustrated in Figure 4.2 can be represented by the following formula:

$$x_l = H_{l([x_0, x_1, \ldots, x_{(l-1)}])} \tag{4.3}$$

The classification layer consists of a global average pooling layer, followed by a fully connected layer and a softmax activation function. During training, DenseNet121 uses a combination of cross-entropy loss and L2 regularization to optimize the network's parameters.

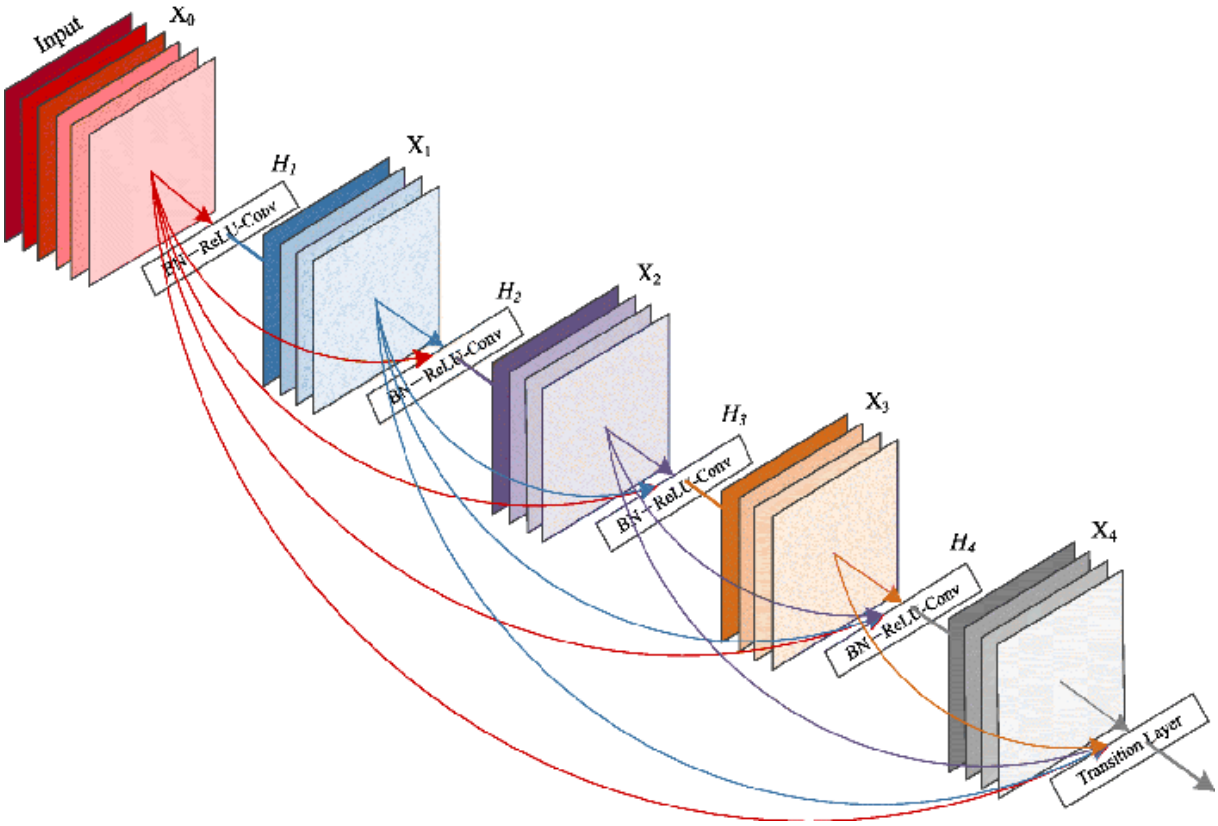

Figure 4.2: Architecture of DenseNet121 [6].

DenseNet121 has achieved state-of-the-art results on a variety of image classification tasks,

including the ImageNet dataset. Its dense connections and feature reuse make it a powerful architecture for image classification and other computer vision tasks.

4.1.4 SimpleCNN

As a benchmark, we compare the more sophisticated methods to a CNN which a much less sophisticated structure which we call SimpleCNN. The SimpleCNN is composed of three convolutions and max pooling layers. The architecture is illustrated in Figure 4.3.

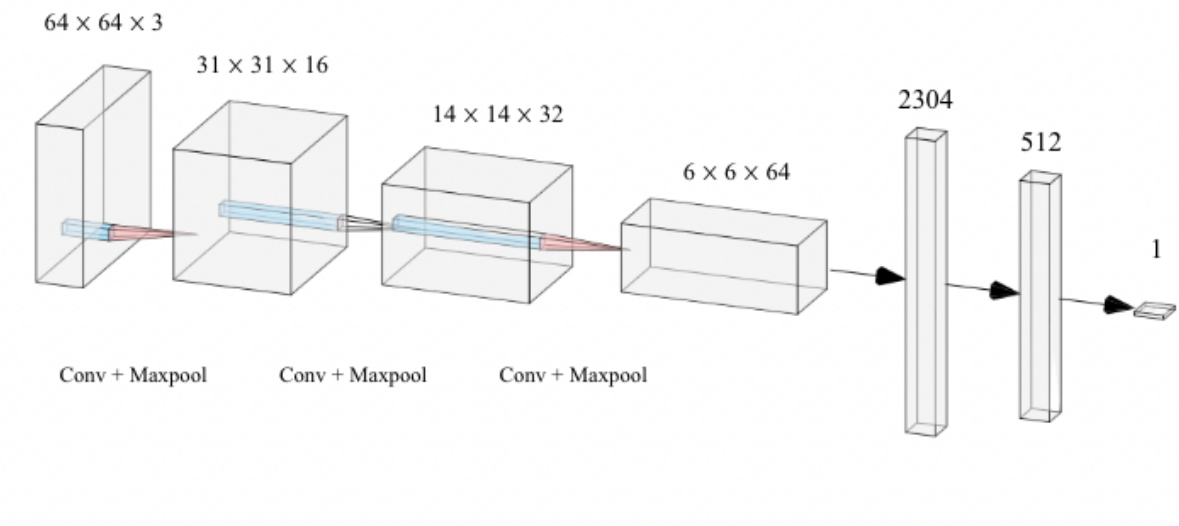

Figure 4.3: Architecture of The SimpleCNN Binary Classifier.

4.2 Flash Score Heatmap(FlaSH)

For the last step of the lightning prediction method, we begin the construction of heatmaps for the whole desired area using the models trained on the 64×64 data. We use patchify[21] to generate 64×64 patches of the whole picture for which the model can predict a probability. Using these patches, we implement a convolution-like task, where the filter is the trained model, the filter size is 64×64 and the stride is 1. By applying the said task, we then generate $(n-63) \times (n-63)$ scores for a $n \times n$ region. We then proceed to create a heatmap of the whole region using the generated scores. An example of such a heatmap is illustrated in Figure 4.4.

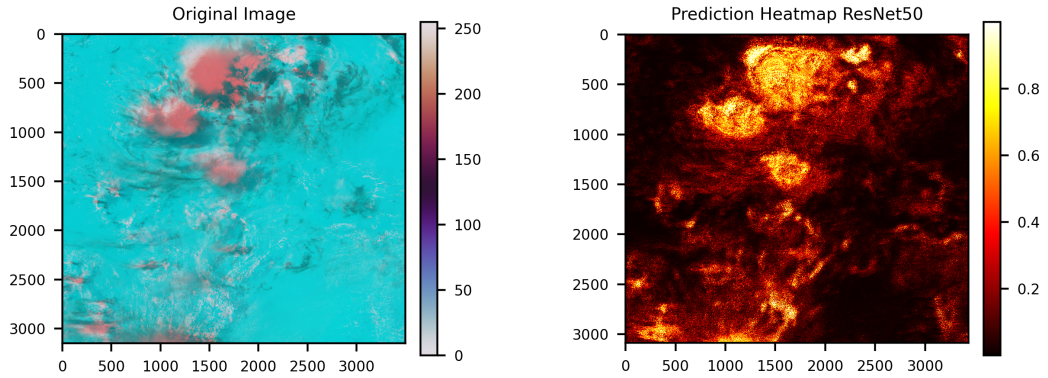

Figure 4.4: On the right side of the figure, the original data of September 20th, 2019 at 15:00 UTC has been shown. On the left side of the figure, we see a heatmap created using ResNet50 scores on the same data for the whole region. We see here that the heatmap created by the model is very similar in structure to the original image.

As shown on the heat map, the strength of the clouds and the combination of the intensities of the three channels is a good indicator of our model's prediction scores. The patterns in the majority of the image seem to be highly similar.
After observing an example of a heatmap created with the scores of one of our models and the similarities with the original image, next to our CNNs we implemented the **Max-Mean(MM)** algorithm. MM works as follows:

- For each 64×64 patch, we take the maximum intensity of each layer(channel).
- The values of channels 13 and 15 are the inverse of channel 2, therefore the lower they are, the stronger the cloud system. Thus, we scale them based on the following formula:

$$max_{ch} = 1 - max_{ch} \tag{4.4}$$

- We then take the average of the three maximum values to create our MM score for the specified patch.

An example of the heatmap created by using the MM approach is illustrated in Figure 4.5

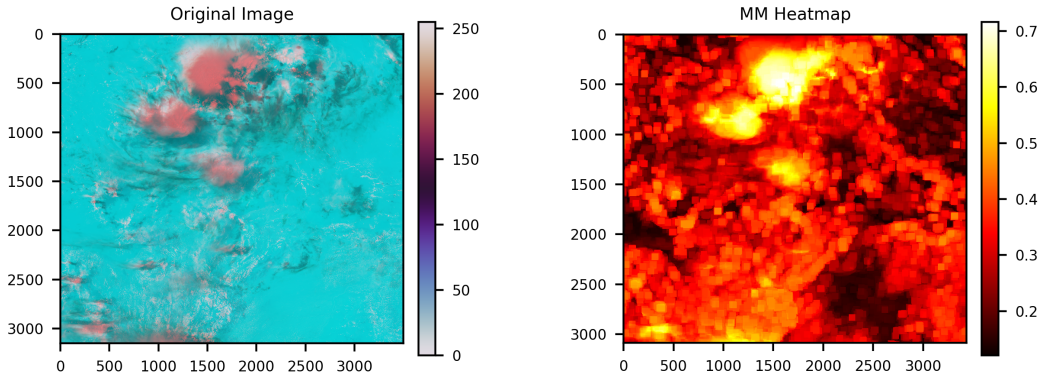

Figure 4.5: On the left side of the figure, the original data of September 20th, 2019 at 15:00 UTC has been shown. On the right side of the figure, we see a heatmap created using MM scores on the same data for the whole region. There are observable parallels in the patterns of the two images, however, the more intense areas in the MM heatmap have lower scores.

Chapter 5

Results

5.1 Model Training

In this section, we present the results of training and testing each of the presented networks on our samples, as well as the produced heatmaps. In Table 5.1 we show the configurations used for training each network. These configurations were chosen by trial and error after many distinct runs.

	No. of Params	lr	Optimizer	Epochs
ResNet50	23.5 Mil	5×10^{-8}	Adam	80
DenseNet	7 Mil	1×10^{-8}	Adam	80
MobileNetV3	1.5 Mil	1×10^{-7}	Adam	100
SimpleCNN	1.2 Mil	1×10^{-7}	Adam	80

Table 5.1: Configurations of The Neural Networks.

We trained each network on the same balanced dataset. Figure 5.1 shows results of the ResNet50 training. We see that, despite initial fluctuations, the model soon converges to near-perfect accuracy and AUC for training and validation. In the case of MobileNetV3 in Figure 5.2, although both validation metrics eventually converge to a value above 0.95, the network needs more time to generalize. As for DenseNet121 in Figure 5.3, the AUC quickly converges to almost 1 in less than 10 epochs. we also see that the validation accuracy is higher than the training accuracy in the case of DenseNet.
In Table 5.2 we see the final validation scores achieved for all the proposed networks. We see that all the scores are extremely high and DenseNet has achieved the highest score during training.

	val_acc	val_auc
ResNet50	**0.98**	0.98
DenseNet	**0.99**	0.98
MobileNetV3Small	**0.99**	0.996
SimpleCNN	0.99	**0.998**

Table 5.2: Final scores achieved for all four networks.

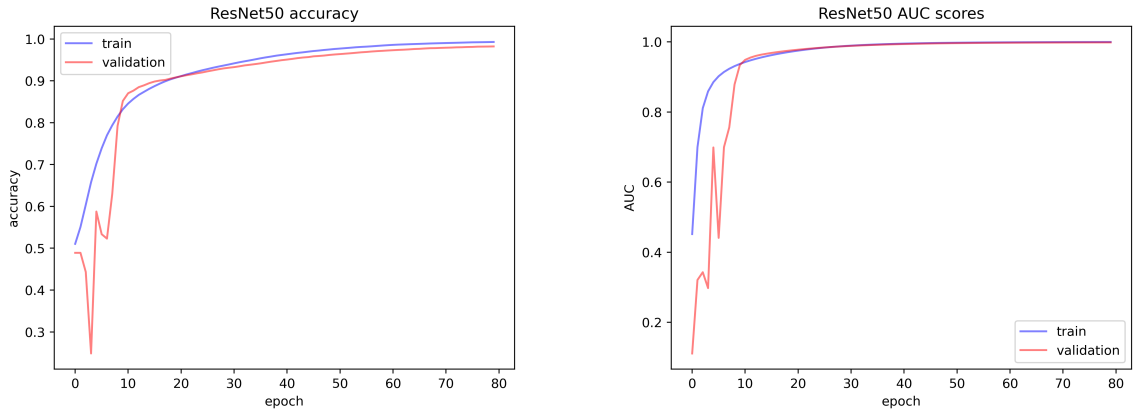

Figure 5.1: Results of training and validation accuracy and AUC of ResNet50. On the left, the training accuracy(blue line) and the validation accuracy(red line) have been demonstrated. On the right, the right panel shows the AUC of the model with the same colors for validation and training. Both metrics converge quickly toward the maximum.

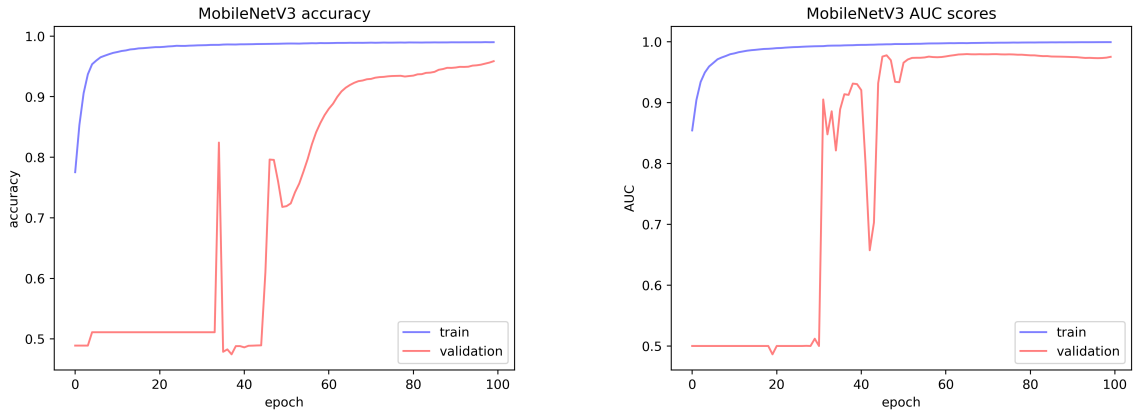

Figure 5.2: Results of training and validation accuracy and AUC of MobileNetV3. On the left, the training accuracy(red line) and the validation accuracy(blue line) have been demonstrated. On the right, we have the AUC of the model with the same colors for validation and training. In the example of this Network, we see that the model first struggles to generalize, but ultimately achieves 0.96 accuracy and 0.97 AUC.

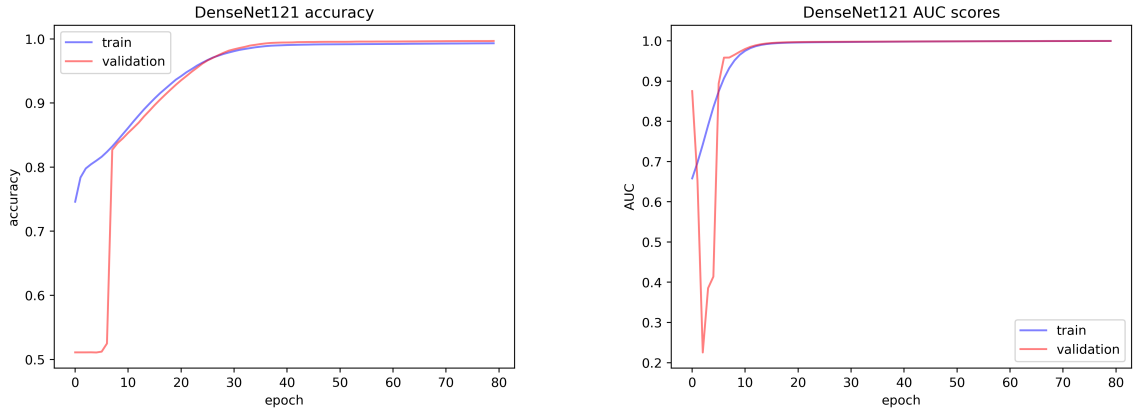

Figure 5.3: Results of training and validation accuracy and AUC of DenseNet121. We see that the AUC quickly rises to the maximum amount in the case of this network.

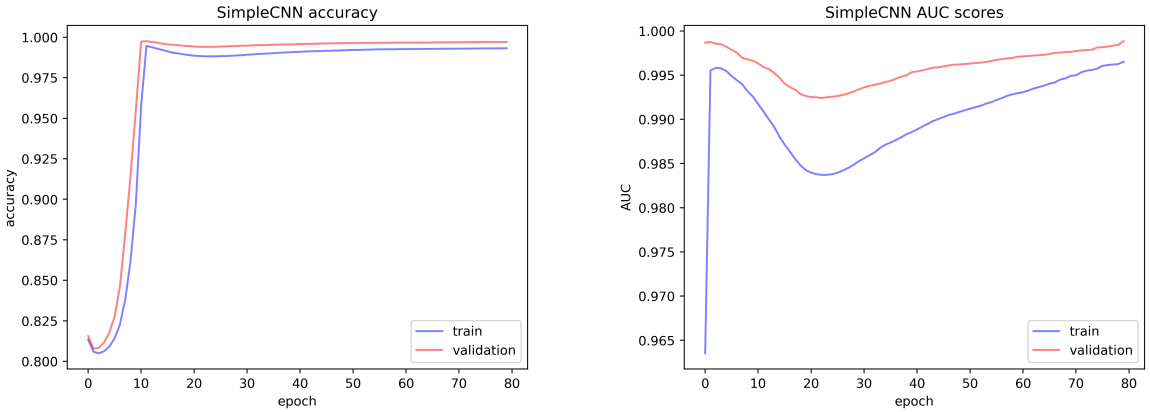

Figure 5.4: Results of training and validation accuracy and AUC of SimpleCNN. Similar to the previous models shown in this study, the network soon converges to its maximum accuracy. We also see here that the validation accuracy and AUC stay consistently higher than training AUC and accuracy.

Due to the strong performance of all models, we experimented with training the model using datasets of varying sizes in order to determine the data size at which the model reaches optimal performance.

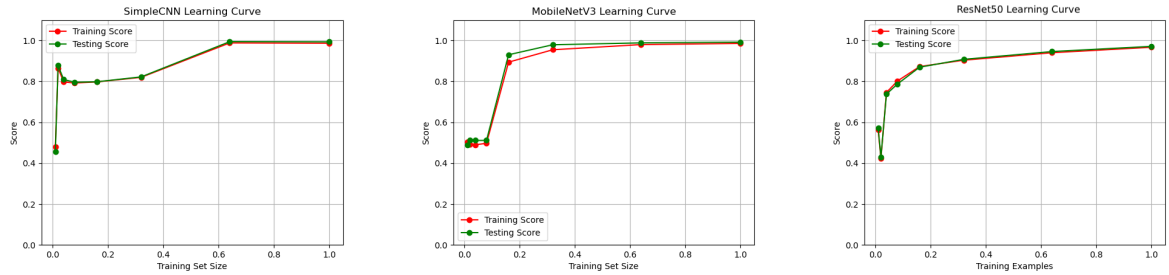

Figure 5.5: This figure shows learning curves for SimpleCNN, MobileNet and ResNet50 from left to right respectively. The horizontal axis represents the size of the training set, while the vertical axis represents the model's performance. In each graph, the red line represents training accuracy while the green line represents validation accuracy. We see that more advanced models converge to high accuracy at 20% of the whole dataset (∼16000 samples) but SimpleCNN requires 60% (∼48000 samples) to converge to the same score.

In Section 2 of this chapter, we present the results of utilizing our best model to create a heatmap of the whole region of interest.

5.2 FlaSH(Flash Score Heatmap)

In this section, we present the outcomes of creating heatmaps using our trained models. In each figure, the MM method is used as a baseline, and the model prediction score heatmap is also shown. Figures 5.6, 5.7, 5.8, 5.9 are instances of the produced heatmaps with different models.

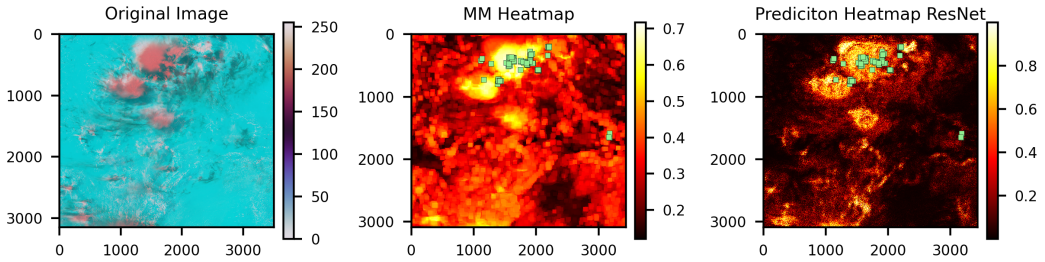

Figure 5.6: The image on the left depicts the original 3-channel cloud data for this location on September 20th, 2019 at 15:00 UTC, where clouds with pinker hues have more intensity. The middle image is a heatmap created by the MM algorithm. . On the right, we see the heatmap created with ResNet50 scores. Flash locations are illustrated as green squares in each heatmap, and lighter colors indicate higher probability of flash occurrence.

Figure 5.6 illustrates that the MM algorithm generates a solid baseline for our heatmaps. While it exhibits a lot of false positives, Has an AUC score of 0.92. ResNet50 obtains the same score, with significantly fewer false positives.

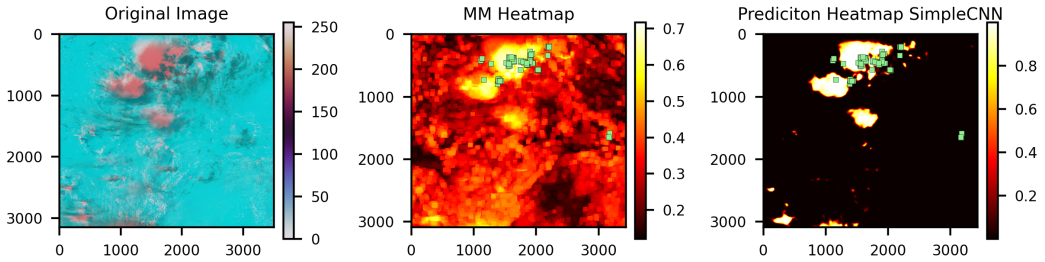

Figure 5.7: Another illustration of a heatmap. On the right is the heatmap that was generated using SimpleCNN scores. Although this heatmap seems to be more accurate than ResNet50, it produced an AUC of roughly 0.89 for the whole area.

Our most effective model generates the FlaSH seen in Figure 5.7. This model generates a heatmap with an AUC score of 0.96. Evidently, it has the lowest number of false positives among the offered models.

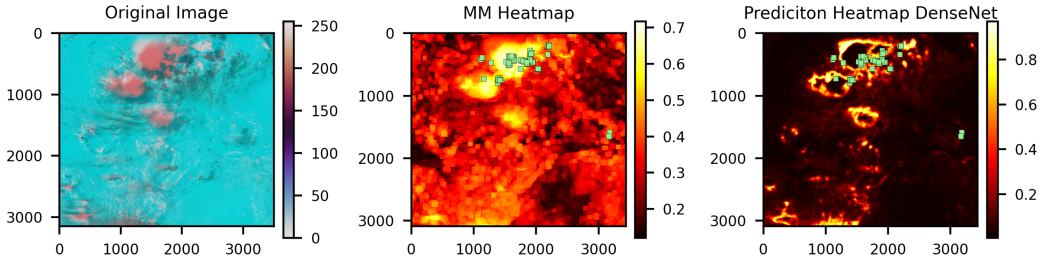

Figure 5.8: On the right, we see the heatmap created with DenseNet121 scores. This heatmap depicts regions with a high chance of flash in previous heatmaps as fully black (areas with a low risk of flash), resulting in an AUC of 0.74. As we hypothesized that this was a result of underfitting the data to the model, we have tested numerous alternative setups to get the best possible heatmap.

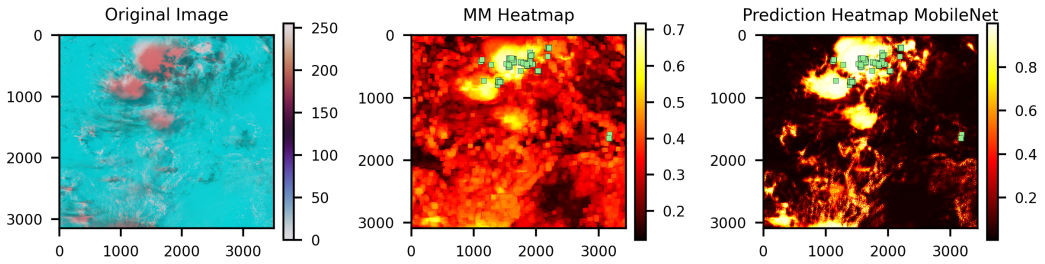

Figure 5.9: On the right, we see the heatmap created with MobileNetV3Small scores.

In addition, we visualized the three distinct underlying channels to determine which had the most impact on the produced heatmap.

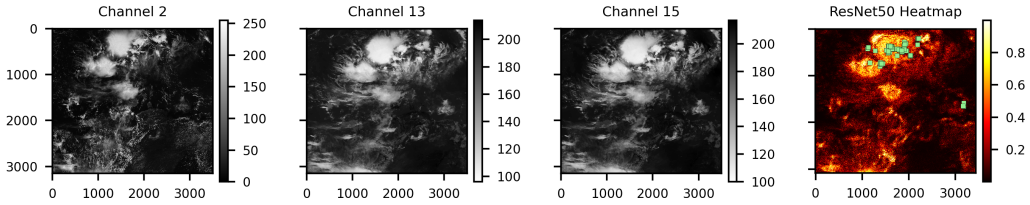

Figure 5.10: Three photos on the left depict the brightness temperatures of the underlying channels for September 20, 2019 at 15:00 UTC, with brighter colors indicating more intensity. ResNet's heatmap is seen on the image to the right. The locations of lightning are shown by green squares.

Chapter 6

Discussion

Contrary to our original notion that our collected data is complicated and contains several distinct patterns, we uncovered throughout model training that this is not the case. In the early stages of training, all four models were able to obtain exceptionally high accuracy on the validation data, indicating that our data was simple for the algorithm to learn. The relationship between these three channels and the occurrence of lightning is very evident. In light of the fact that we did not integrate variables such as time, the quantity of lightning, lightning area, etc., it may in hindsight not be so surprising that the model can easily address this problem.

Our results also reveal that the intensity of the three channels provides a solid starting point for our prediction job. MM show promising results by achieving AUC scores higher than 0.90. This helped us understand why neural networks were so adept at recognizing visual patterns.

Furthermore, by examining the learning curves of our networks, we can show that it is feasible for individual networks to reach high accuracy with fewer dataset samples. In addition, we see that MobileNet requires less data than ResNet to converge to extremely high accuracy.

Additionally, we found that the model's complexity impacts the level of information in each heatmap. We see that the greater the number of model parameters, the more specific the prediction scores, and therefore, the more detailed the heatmaps generated. This does not necessarily suggest that the model is predicting the locations of flashes more correctly. In the case of DenseNet, as our second-most sophisticated model, the resulting heatmap is inaccurate.

We discovered that although the heatmap created by ResNet appears to be a more detailed representation of the cloud system, but SimpleCNN produces the most accurate FlaSH. With the AUC score of 0.96 for the whole region, we can confidently say that this model is able to predict the lightning locations accurately, considering that we are looking at 8km $\times$ 8km areas for the AUC calculation. More sophisticated neural networks have not been investigated, as the explored models already achieved extremely high accuracy.

While most of our heatmaps seem to be reliable, there appears to be some false positives,

indicating that the model predicts lightning when none exists. This may influence decision making. We think this is due to the fact that each of our samples were obtained from a 64×64 (roughly 32×32 km) pixel region, and we designated each sample as a flash sample if there was just one instance of lightning in that image. This leads the model to infer that every intense cloud structure over a large region is a flash sample, which is not always the case. Moreover, the simplicity of the data may lead the more sophisticated models to be somewhat overfitted. We infer this based on the fact that the SimpleCNN heatmap has the lowest rate of false positives among all heatmaps. A basic method for resolving this problem would be to consider only regions with a chance of flash occurrence averaging more than 0.80. Hyperparameter optimization and additional regularization of the models are also possibilities to consider.

Chapter 7

Conclusion

Using ABI and GLM data files for a selection of days of the year 2019, to predict the occurrence of lightning over an area in the Caribbean, a dataset of 64×64 pixel patches for training networks and binary image classification was constructed.

The classification of our small samples was shown to be a very straightforward operation for both basic and sophisticated neural networks. We can extract sample classification from a single flash event, which may make detecting cloud patterns for the network simple. Therefore, we see that all networks obtain very high accuracy on our validation data in the early stages of training.

Using our trained models, we have demonstrated a novel approach by producing heatmaps of the area(FlaSH). We noticed that, despite the fact that model complexity increases the details included in a heatmap, these details might impair the accuracy of the heatmap by increasing the number of false positives. We observed that the SimpleCNN produced the most accurate heatmap in terms of properly forecasting the locations of positive flashes and that the use of more sophisticated models is unnecessary.

We noticed that, despite the fact that all models acquire an exceptionally high score on the test set, the heatmaps generated by the models do not share the same structure or AUC score. Given that we eliminated all possibilities for data leakage, overfitting, and programming faults, determining the cause of this difference was beyond the scope of this study.

In conclusion, FlaSH is a reliable tool for lightning prediction using satellite data. The FlaSH created by the SimpleCNN model, appears to be the most accurate heatmap for lightning location prediction 35 minutes ahead of time.

Chapter 8

Future Work

While training the model on 64 $\times$ 64 pixel patches seems to be a simple task for the models, it is possible to conduct more tests using patches with larger dimensions. Using additional (all) wavelength channels for sample generation and model training is an additional topic that may be investigated. Furthermore, the potential of training the model with additional data, such as flash area, flash extent density, etc., may be investigated. This issue may also be expressed as a time series problem.

We created FlaSHs of multiple ABI files at different times in this research. With more computational resources, more investigation into the accuracy of this approach on a bigger dataset of ABI files can give further insight into the reliability of the heatmaps.

Bibliography

[1] Alexander P. Trishchenko. "Solar Irradiance and Effective Brightness Temperature for SWIR Channels of AVHRR/NOAA and GOES Imagers". In: *Journal of Atmospheric and Oceanic Technology* 23.2 (2006), pp. 198–210. DOI: 10.1175/JTECH1850.1. URL: https://journals.ametsoc.org/view/journals/atot/23/2/jtech1850_1.xml.

[2] Edward R. Mansell, Conrad L. Ziegler, and Donald R. MacGorman. "A Lightning Data Assimilation Technique for Mesoscale Forecast Models". In: *Monthly Weather Review* 135.5 (2007), pp. 1732–1748. DOI: https://doi.org/10.1175/MWR3387.1. URL: https://journals.ametsoc.org/view/journals/mwre/135/5/mwr3387.1.xml.

[3] Jia Deng et al. "Imagenet: A large-scale hierarchical image database". In: *2009 IEEE conference on computer vision and pattern recognition*. Ieee. 2009, pp. 248–255.

[4] B. Lynn and Y. Yair. "Prediction of lightning flash density with the WRF model". In: *Advances in Geosciences* 23 (2010), pp. 11–16. DOI: 10.5194/adgeo-23-11-2010. URL: https://adgeo.copernicus.org/articles/23/11/2010/.

[5] Kaiming He et al. "Deep Residual Learning for Image Recognition". In: *Proceedings of 2016 IEEE Conference on Computer Vision and Pattern Recognition*. CVPR '16. Las Vegas, NV, USA: IEEE, June 2016, pp. 770–778. DOI: 10.1109/CVPR.2016.90. URL: http://ieeexplore.ieee.org/document/7780459.

[6] Gao Huang, Zhuang Liu, and Kilian Q. Weinberger. "Densely Connected Convolutional Networks". In: *CoRR* abs/1608.06993 (2016). arXiv: 1608.06993. URL: http://arxiv.org/abs/1608.06993.

[7] Nicholas Read, Thomas J. Duff, and Peter G. Taylor. "A lightning-caused wildfire ignition forecasting model for operational use". In: *Agricultural and Forest Meteorology* 253-254 (2018), pp. 233–246. ISSN: 0168-1923. DOI: https://doi.org/10.1016/j.agrformet.2018.01.037. URL: https://www.sciencedirect.com/science/article/pii/S0168192318300376.

[8] Timothy Schmit et al. "Applications of the 16 spectral bands on the Advanced Baseline Imager (ABI)." In: *Journal of Operational Meteorology* 06 (June 2018), pp. 33–46. DOI: 10.15191/nwajom.2018.0604.

[9] Ronald Holle, Mary Ann Cooper, and Managing. "Overview of Lightning Injuries Around the World". In: July 2019.

[10] Andrew Howard et al. "Searching for MobileNetV3". In: *CoRR* abs/1905.02244 (2019). eprint: 1905.02244. URL: http://arxiv.org/abs/1905.02244.

[11] Amirhossein Mostajabi et al. "Nowcasting lightning occurrence from commonly available meteorological parameters using machine learning techniques". In: *npj Climate and Atmospheric Science* 2 (Nov. 2019), p. 41. DOI: 10.1038/s41612-019-0098-0.

[12] Dominick V Speranza. "Lightning Prediction Using Recurrent Neural Networks". In: *Theses and Dissertations* (2019). URL: https://scholar.afit.edu/etd/2317.

[13] John L. Cintineo et al. "A Deep-Learning Model for Automated Detection of Intense Midlatitude Convection Using Geostationary Satellite Images". In: *The 1st NOAA Workshop on Leveraging AI in the Exploitation of Satellite Earth Observations Numerical Weather Prediction* (2020), pp. 2567–2588. DOI: https://doi.org/10.1175/WAF-D-20-0028.1e. URL: https://journals.ametsoc.org/view/journals/wefo/35/6/waf-d-20-0028.1.xml.

[14] Yaseen Essa, Ritesh Ajoodha, and Hugh Hunt. "A LSTM Recurrent Neural Network for Lightning Flash Prediction within Southern Africa using Historical Time-series Data". In: Dec. 2020, pp. 1–6. DOI: 10.1109/CSDE50874.2020.9411544.

[15] Seung-Hyun Moon and Yong-Hyuk Kim. "Forecasting lightning around the Korean Peninsula by postprocessing ECMWF data using SVMs and undersampling". In: *Atmospheric Research* 243 (May 2020), p. 105026. DOI: 10.1016/j.atmosres.2020.105026.

[16] Kanghui Zhou et al. "A Deep Learning Network for Cloud-to-Ground Lightning Nowcasting with Multisource Data". In: *Journal of Atmospheric and Oceanic Technology* 37.5 (2020), pp. 927–942. DOI: 10.1175/JTECH-D-19-0146.1. URL: https://journals.ametsoc.org/view/journals/atot/37/5/jtech-d-19-0146.1.xml.

[17] Luqman Ali et al. "Performance Evaluation of Deep CNN-Based Crack Detection and Localization Techniques for Concrete Structures". In: *Sensors* 21 (Mar. 2021), p. 1688. DOI: 10.3390/s21051688.

[18] John L. Cintineo, Michael J. Pavolonis, and Justin M. Sieglaff. "ProbSevere LightningCast: A Deep-Learning Model for Satellite-Based Lightning Nowcasting". In: *Weather and Forecasting* 37.7 (2022), pp. 1239–1257. DOI: 10.1175/WAF-D-22-0019.1. URL: https://journals.ametsoc.org/view/journals/wefo/37/7/WAF-D-22-0019.1.xml.

[19] Yaseen Essa et al. "Deep Learning Prediction of Thunderstorm Severity Using Remote Sensing Weather Data". In: *IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing* 15 (Jan. 2022), pp. 1–1. DOI: 10.1109/JSTARS.2022.3172785.

[20] David A. Yuen et al. "Under the surface: Pressure-induced planetary-scale waves, volcanic lightning, and gaseous clouds caused by the submarine eruption of Hunga Tonga-Hunga Ha'apai volcano". In: *Earthquake Research Advances* 2.3 (2022), p. 100134. ISSN: 2772-4670. DOI: https://doi.org/10.1016/j.eqrea.2022.100134. URL: https://www.sciencedirect.com/science/article/pii/S2772467022000227.

[21] dovahcrow. *patchify*. URL: https://github.com/dovahcrow/patchify.py.

[22] Scott Lindstrom. *The Split Window Difference as a measurement of Atmospheric Moisture*. URL: https://cimss.ssec.wisc.edu/satellite-blog/archives/23702#:~:text=The%5C%20brightness%5C%20temperature%5C%20difference%5C%2C%5C%20nicknamed,in%5C%20moisture%5C%20in%5C%20clear%5C%20skies.%5C.

[23] NOAA. *ABI BANDS QUICK INFORMATION GUIDES*. URL: https://www.goes-r.gov/mission/ABI-bands-quick-info.html. (accessed: 10.10.2022).

[24] NOAA. *ABI RadF Data on Amazon AWS*. URL: https://home.chpc.utah.edu/~u0553130/Brian_Blaylock/cgi-bin/goes16_download.cgi.

Appendix A

Days of Data

Table A.1: List of the days that were used for sample creation, divided into training and validation.

2019 Days of The Year	
Training	Validation
108	307
167	308
192	310
194	311
210	325
240	345
258	346
262	355
263	
264	
265	
266	
267	
268	
269	
271	
273	
274	
275	
276	
277	
278	
279	
280	
281	
282	
284	
285	
287	
291	
292	
295	