Identifying and forecasting thunderstorms using weather radar data and machine learning

Master's Thesis, 20.11.2023

Author:

Joona Huttunen

Supervisors:

Antti Mäkelä (Finnish Meteorological Institute)
Seppo Pulkkinen (Finnish Meteorological Institute)
Arttu Miettinen

UNIVERSITY OF JYVÄSKYLÄ
DEPARTMENT OF PHYSICS

Abstract

Huttunen Joona
Identifying and forecasting thunderstorms using weather radar data and machine learning
Master's thesis
Department of Physics, University of Jyväskylä, 2023, 59 pages.

Methods for nowcasting lightning using weather radar data were developed using machine learning models. Reflectivity was selected as the main feature for the prediction. The purpose was to examine if machine learning applications could be used to nowcast thunderstorms with minimal data sets. The emphasis was to find out a model which is based on binary image classification and doesn't require large sets of training data to work sufficiently. Convolutional neural network was the first choice. Accuracy for the model was 0.83. Another approach was made using random forest model. Precision for class 0 (no lightning) was 0.52, and for class (recorded lightning) 1, 0.90 and with total accuracy of 0.88 To improve the sets more features should be used and possibly larger data sets.

Keywords: Atmosphere, Lightning, Nowcasting, Machine learning, Classification

Tiivistelmä

Huttunen Joona
Identifying and forecasting thunderstorms using weather radar data and machine learning
Pro gradu -tutkielma
Fysiikan laitos, Jyväskylän yliopisto, 2023, 59 sivua

Salamoinnin ennustamiseen säätutkatietojen avulla kehitettiin koneoppimismenetelmiä. Tarkoituksena oli tutkia, voidaanko koneoppimista käyttää salamoinnin ennustamiseen pienillä datamäärillä. Pilvien heijastuvuus valittiin tärkeimmäksi selittäväksi tekijäksi. Tavoitteena oli löytää malli, joka pohjautuu kuvantunnistukseen. Konvoluutioneuroverkko oli ensimmåinen valinta. Tarkkuus mallille (accuracy) oli 0.83. Toisena lähestymistapana käytettiin random forest -mallia, joka soveltui laskentateholta nopeammin käytettyihin datamääriin. Tarkkuus luokan 0 (ei osumaa) osalta oli 0,52 ja luokan 1 (osuma määrätyssä alueessa) osalta 0,90. Tulosten parantamiseksi olisi käytettävä enemmän myrskyn kehityksen kannalta merkittäviä muuttujia ja suurempia datamääriä.

Avainsanat: rajuilmat, koneoppiminen, kuvantunnistus

Prologue

This study originated from my own idea but was made possible the Finnish Meteorological Institute. I would like to thank the people at the FMI for their help and insights on the subject. I would also like to thank to my supervisor Arttu Miettinen for his support.

Jyväskylä 2023

Joona Huttunen

Contents

1 Introduction

Storms and thunderstorms occur all over the globe. During summertime in Finland thunderclouds occur almost daily. In comparison to the global average, the annual lightning rates are low in Finland [1]. The clouds can be singular and small in magnitude but it's not uncommon for them to consist of multiple clouds clustered together. These clusters can be hundreds of kilometers in size. Three main contributing factors in the development of rain- and thunderclouds are humidity, instability of air convection, and updraft. [2].

Thunderstorms (thunderclouds) and their development can be forecast using weather radars. Weather radars can collect data about the amount of rain in a specific cross-section of a cloud. They can also get information about the structure of clouds and the small particles that they consist of, ie. rain droplets and snowflakes. By using statistics from previous years and present weather data we can create a nowcast (a forecast of immediately expected weather conditions) about the current weather.

There has been a growing trend in using large sets of data and machine learning principles for weather forecasting in recent years. Machine learning offers possibly greater accuracy in forecasts and computationally faster results. It also excels when using very large data sets. [3]

The purpose of this thesis is to examine the usage of multiple weather radar data sources and machine learning methods to create accurate nowcasts of lightning strikes. Another goal is to examine the coefficient of determination and which variable would be most suitable for nowcasting. The information acquired from weather radar data has greater temporal and spatial resolution compared to numerical weather prediction models. The overall hypothesis is that by combining radar data and machine learning methods we can possibly create more precise lightning forecasts. There have been only few recent studies on this subject and it hasn't been done using Finnish weather data.

2 Theory of thunderstorms

Thunder is a weather phenomenon in which electrical charges formed in clouds discharge as a lightning strike. During the discharge, the temperature of the air around the electric arc increases significantly. The air expands while creating a shock wave. This shock wave and the sound created by it is commonly known as 'thunder'. Thunderstorms form in convective clouds, in which the hot moist air ascends upwards and condenses to water. A typical feature of a thundercloud is its height, which is usually the same vicinity as its diameter. Vertical winds are another typical feature of thunderclouds; the average wind speed in their vicinity is 10 m/s-50 m/s. [4]

2.1 Electricity in the atmosphere

During the nineteenth century, it was discovered that air is continuously ionized. This discovery led to the conclusion that the Earth's charge resided in the upper atmosphere while being practically constant. [5] The region of interest is presented in the fig. 1

The electrical currents and processes that occur in the earth's atmosphere are referred to as the electric global circuit. It is created mostly by approximately 1 500 tropical thunderstorms occurring at any given moment, which each one charging the upper atmosphere with a current of 1A [6]. The ionosphere acquires a potential of 300 kV with respect to the ground. The lower atmospheric air has an electrical conductivity of 20 fS/m. The conductivity follows an exponential growth pattern of height, increasing by a factor of ten with every 10 km increase in altitude. In the upper regions, near the ionosphere, the conductivity increases due to the increase in the number of free electrons and the effect of Earth's magnetic field, and the number of ions in circulation. The electric field becomes anisotropic. At 80 km the conductivity is approximately 1 S/m [7]. The current returning to earth from the ionosphere experiences a resistance, described by the column resistance, R_c in the Fig. 1 Cosmic rays and radiation from radon gas cause ionization in the atmosphere which also contribute to the global circuit.

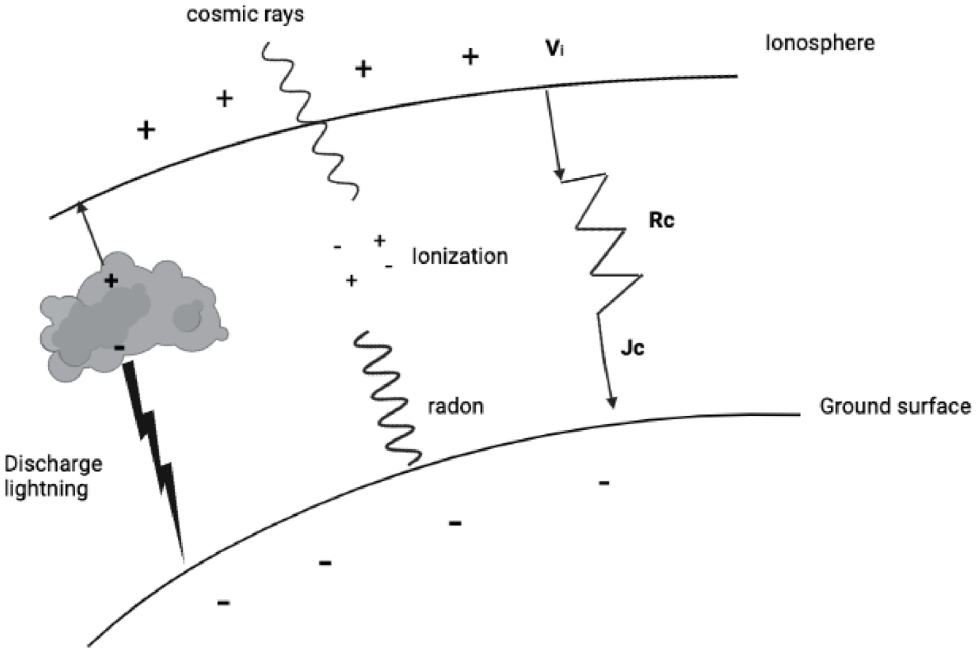

Figure 1. Visualization of the electric global circuit. The potential difference can be seen from the negative and positive signs, electrical discharging and the contribution on cosmic rays and radon.

The magnitude of the electric field near the ground on fair weather is on average 100 V/m. In fair weather, there is a flow of electric current from the atmosphere to the ground, with an average current density J of $2.8\,\mathrm{pA/m^2}$ in southern Finland, and an average of approximately $2\,\mathrm{pA/m^2}$ worldwide [8] Ohm's law usually governs the relationship between current density J, conductivity σ, and electric field E, at least approximately,

$$J = \sigma E. \tag{1}$$

The land surface shapes affect the magnitude of the electric field, but the changes need to be quite drastic to have any meaningful effect on lightning strikes [8].

2.2 Formation of a thunder cell

The development of convective clouds requires a favorable change in air temperature in the troposphere. A significant drop in temperature can shift the temperature distribution in the cloud to an unstable state; if such an event happens it can cause strong vertical air currents. The term unstable refers to the lapse rate occurring in the cloud, ie., in "normal" conditions the temperature decreases as you go higher in the atmosphere. In unstable conditions, the warm air can rise so that it is cooler than its surroundings. The center of current in which the air flows upwards is called a convection cell. As the convection cell develops, the cloud grows in height, and at the same time the area of the convection cell increases. If there is enough humidity in the cloud, the formation of thunderclouds is possible. Condensing water vapor has a higher temperature which slows down the cooling process of the air while increasing the vertical air flow. The condensing droplets form a cumulus cloud. As the cumulus grows large enough, it transforms into cumulonimbus. The upper region of cumulonimbus is called the anvil and there small ice crystals are formed. Favorable circumstances can lead to electric charging in the cumulonimbus which can eventually discharge as lightning. This type of cumulonimbus is more commonly called a thundercloud. On average the area of a thunder cell is a few kilometers in size. Thunder cells can cluster together forming larger thunderclouds with diameters up to 40 - 50 km. It is nearly impossible to distinguish separate thunder cells in such clusters. [8, 9]

2.3 Electric charging of a thunder cell

As mentioned previously the atmosphere under 100 km is electrically conductive due to tropical thunderstorms, cosmic rays, and the ions caused by the Earth's natural radioactivity. The single most contributing factor to the conductivity of lower atmosphere above 1 km are the small ions whose diameter is in the vicinity of 0.1 nm - 1 nm. Ions can collide with one another and doing so form neutral gas molecules. Ions can also stick to aerosol or water particles while charging or neutralizing them. [9] The electrical mobility of small ions is $k = 1.2 \cdot 10^{-4}$ m^2/Vs and the conductivity λ is the sum of positive- and negative-ion components:

$$\lambda = \lambda^+ + \lambda^- = ke(n^+ + n^-) \tag{2}$$

where n^+ and n^- are the the small ion densities and e is the elementary charge.

An essential factor in the charging of a thundercloud is the physical properties of water in its frozen state. There are many theories about the effect of water and ice crystals in the electrical charging of the thundercloud. The most commonly accepted model is based on the collision of ice crystals and water droplets. Convection currents convey the subcooled water upwards so that it forms ice crystals. These ice crystals collide with graupel in the upper regions of the cloud. During a neutral collision, the electrical charge is transferred from one particle to another; the reasons behind this phenomenon are suggested to be the crystal structure of ice and the temperature change. Induction collision happens when there is a stronger charge on one of the colliding particles. Charge transfer occurs in the collision of graupel and droplets and the particle with greater charge falls towards the ground in the direction of the electric field. Due to the internal charges the falling particle is polarized. The charges in the collision point are neutralized and only a net charge is left. [5, 10, 11] It is worth mentioning that snow itself is a fairly poor conductor but due to the impurities and contaminants, it can acquire a certain level of conductivity [12].

The rate of change of charge Q of a large particle (snow grain) is proportional to the collision frequency v between it and small particles (ice crystals and cloud droplets)

$$\frac{\partial Q}{\partial t} = v[Q' + \beta E + \gamma Q] \tag{3}$$

The charge transferred from the small to the large particle in a single collision is *Q'*. The coefficient β describes the effectiveness of the electric field E in causing inductive charging and the coefficient γ indicates that the rate of charge dissipation is proportional to the charge Q itself. The collision frequency is approximately

$$v = \pi R^2 wn \tag{4}$$

Where R is the radius of the large particle, w is the velocity difference between the small and large particles and n is the density of small particles. Coefficient v is then approximately 4s^{-1} The induction coefficient is approximately

$$\beta \approx (4\pi^3/3)\epsilon_0 r^2 \tag{5}$$

where ϵ is the permittivity and r is the radius of the small particle. Coefficient β is approximately 1.5×10^{-19}Cm/V [8].

The sign of charge is affected essentially by the prevailing temperature. Ice crystals charge negatively and the graupel positively if the temperature is below -15°C. If the temperature is higher then the signs are reversed. For this reason, a typical thundercloud is polarized so that the top of the cloud is positively charged and the bottom is negatively charged. Alternatively, the top of the cloud can be positively charged, the middle part negatively, and then the bottom positively charged; this is most common in fully developed Cumulonimbus incus (anvil clouds). The charge structure of thunderclouds is presented visually in Fig. 2

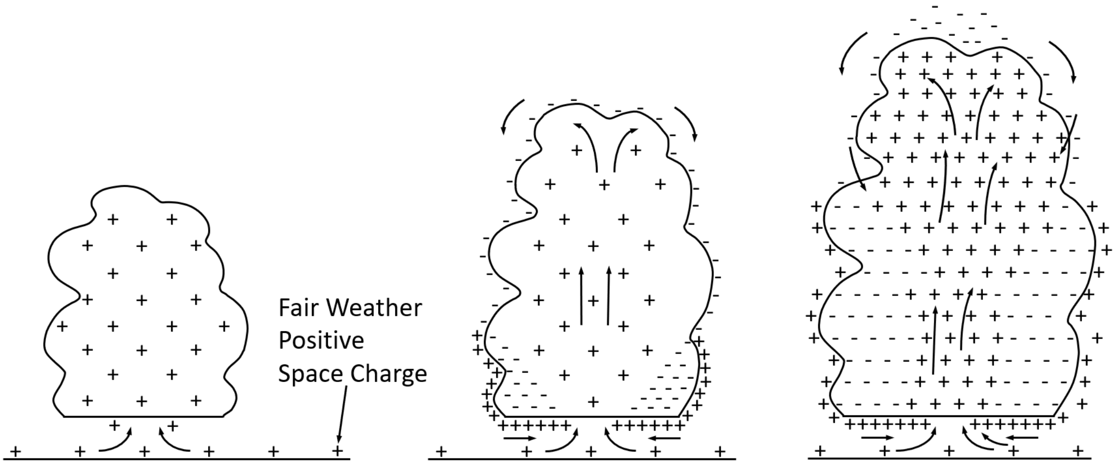

Figure 2. Structure of electrical charge distribution in thunderclouds, adapted from MacGorman [13].

2.4 Lightning

Lightning is the term used to describe the discharge of electrification of thunder cells. A detailed review of the discharge is quite difficult due to its complex nature. This section focuses on a general description of different kinds of lightning discharges. The simplest physical model of thunderclouds can be represented with a dipole model. The electrical charging of thunder cells can be reduced so that the upper part of the cloud is positive and the middle/lower part is negative, as explained in the previous chapter. As the negative lower part of the cloud is physically closest to ground it is induced with a positive voltage. When the voltage grows sufficiently enough between the ground and the cloud, the breakdown threshold is exceeded and the current is discharged as lightning. The discharge happens through a so-called lightning channel. In this specific channel, the gases of the atmosphere are ionized and the conductivity is increased. During the discharge, the gases are heated up to 30 000°C and expand explosively. This can be observed as bright glowing lightning and a loud thunder. [14]. Lightning can strike from cloud-to-ground (CG), from ground-to-cloud (GC), or cloud-to-cloud (CC). Because of this lightning can be classified as cloud lightning and ground lightning. Different types are presented in Fig. 3

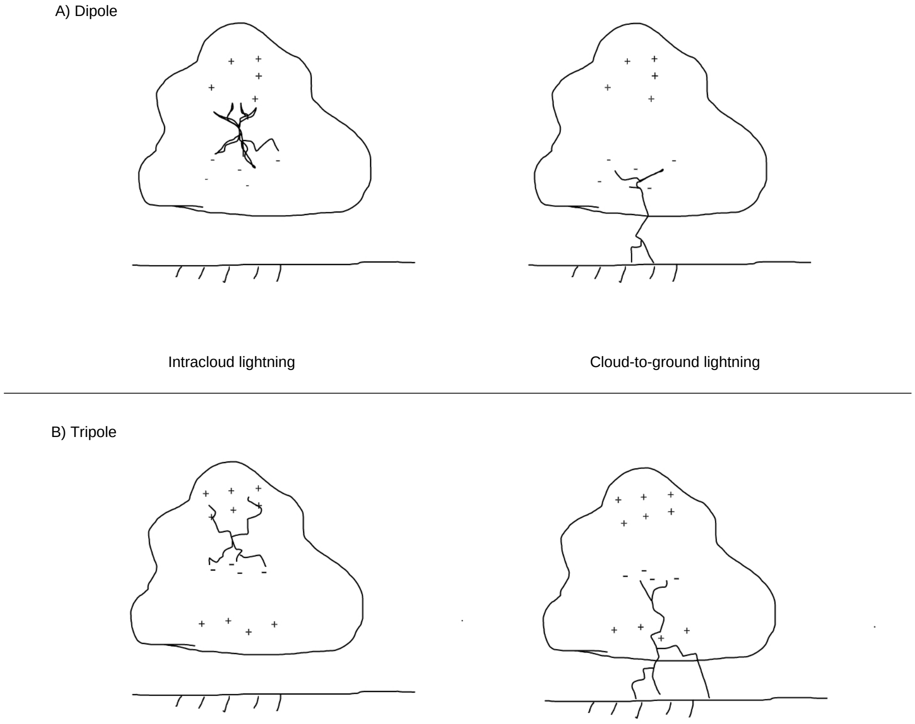

Figure 3. Different kind of lightning types presented, the upper row represents CC and CG lightning starting from the left, the lower row in the same order but with a tripole charge distribution.

2.4.1 Negative ground lightning

Negative ground lightning is estimated to be the most common CG lightning type. This type of lightning transfers a negative charge from the cloud to the ground. About 90 % of CGs are negatively charged [9]. The discharge usually begins with a leader stroke from the cloud which induces a return stroke from the positive side (ground). The return stroke neutralizes the leader when the two pairs connect. There is photographic evidence that the negative leader stroke travels towards the ground with intermittent steps. The time interval of these steps is approximately 10-100 μs and the length is 5-200 m. [15] Lightning flash consists of one or more strokes which happen in the previously introduced lightning channel.

Lightning can occur from the same lightning channels or fork into two or more channels. This is affected by the location of lightning and the physical shape of the surrounding area in the ground. Negative ground lightning can also initiate from the ground when the discharge begins from the ground level. This is known as upward negative lightning.

2.4.2 Positive ground lightning

Positive ground lightning accounts for about 10 % of CG lightning. Due to its rarity, there are relatively few studies about positive lightning. There are few theories on how they occur. One is that the charge structure of the thundercloud is reversed so that the bottom region is positive and the top region negative. Another one is that the charge structure remains the same as usual so that the positive charge remains at the top, but the geometry of the cloud is tilted so that there is a clear path to the ground from the top. These are visualized in the Fig. 3. [16] The leader and return stroke pairing happens the same way as in the negative CG lightning. The notable difference is that positive lightning usually has only one stroke, whereas the negative lightning usually has multiple. Positive lightning leader strokes can move continuously or in a stepped fashion. A final note on the interesting qualities of positive lightning is that they hold the record of lightning currents (near 300 kA). [17] Positive lightnings are also more common during winter thunderstorms and and thunderstorms formed over forest fires or contaminated by other smoke. [18, 19] There are many theories for these correlations, for example, that winter clouds might have a dominantly positive charge [20]. There is yet to be an unambiguous explanation.

2.4.3 Cloud lightning

Cloud lightning (also known as intracloud lightning) or cloud discharge is a type of lightning in which there is no contact made with the ground. Intracloud lightning (IC) covers approximately 75 % of all lightning. Williams also states that typically 10 or more cloud flashes occur before the first cloud-to-ground lightning [21]. The upper dipole region experiences charge separation due to differential motions and collisions between ice particles, resulting in the phenomenon known as IC lightning. IC lightnings dominate the early stages and are evidently correlated with the upward development of the thundercloud. When the upward draft settles down, the IC lightning rates reduce. CG lightning typically initiates 5-10 minutes after IC lightning peak. Cloud lightning is less studied than ground lightning. This is due to the difficulty of getting photographic records of cloud lightning and the inability to measure direct currents in the clouds during discharge events. Modern satellite imagery has helped with the study of IG lightning. [22]

3 Techniques of measurement and assessment of thunderstorm related data

3.1 Weather radars

Weather radars emit short bursts of electromagnetic energy into the atmosphere, typically in the form of microwave radiation, to detect the presence of hydrometeors - which refer to any form of atmospheric moisture that condenses or precipitates into solid or liquid particles. The size of hydrometeors depend on the surrounding conditions. Fog has diameter of 50 µm, whereas hail has around 1-15 mm. When these bursts hit a hydrometeor such as rain, hail, or snow, Rayleigh scattering occurs, causing some of the energy to be reflected to the radar receiver. Rayleigh scattering occurs when the wavelength of the energy is greater than the diameter of the hydrometeor particles, and the different wavelengths can help identify particles of different sizes. [23, 24]

Weather radar systems are usually composed of five components: the transmitter, antenna, radar processor, receiver, and display system. The transmitter produces electromagnetic pulses, which are subsequently transmitted into the atmosphere by the antenna and it also receives the reflected pulses. The antenna has the capability to perform a full 360-degree horizontal rotation and conduct atmospheric scans at different elevation angles. Weak signals are amplified by the receiver, and the radar processor analyzes the received data. This data can be stored in a preferred format or presented directly on a display.

The three main frequency bands used in weather radar are S, C, and X. S-band radar has a longer wavelength and can detect rainfall up to 300 km away, while C-band radar has a medium range and can measure up to 200 km. X-band radar has the smallest wavelength and is can be used for detecting precipitation up to a range of 50 km. There are two more bands which are less common in weather radars. L-band radar is used for studying clear air turbulence and has a frequency of 1-2 GHz, while K-band radar is more sensitive than the X-band and is used for heavy rain detection, with a frequency of 27-40 GHz and 12-18 GHz. Rayleigh scattering becomes less accurate for hydrometeor diameters larger than 2-3 cm when using the X-band. [25]

Except for the northernmost Lapland, radar measurements cover nearly all of Finland as long as the data is taken out to the maximum measurement range of 250 km. The network of radar measurements was created by strategically placing radars, taking advantage of the country's relatively flat topography. This approach has resulted in a network with minimal gaps due to beam blockages caused by terrain, buildings, or other obstructions. [26]

The radar used by the Finnish Meteorological Institute (FMI) is VRM200 by Vaisala. The radar uses dual-polarization technology, which transmits and receives both horizontal and vertical polarization signals. This capability enables the system to obtain additional information about the shape, size, and type of precipitation particles. The technical specifications can be found on the data sheet [27].

3.2 Precipitation analysis based on weather radar data

Precipitation estimates can be obtained using different measurement techniques such as weather radar networks, and meteorological satellites. Rain gauge networks measure the rainfall directly.

The dBZ radar reflectivity product displays the intensity of echoes in decibels. The color scheme used to represent the intensity of echoes varies depending on the software or user preference. Typically, green or light blue indicates weak echoes related to light rain with dBZ values around 20. Yellow represents moderate precipitation with values approximately 35 dBZ, while red indicates heavy precipitation with values around 50 dBZ. The highest reflectivity values, approximately 65 dBZ, indicate the presence of hail mixed in with liquid hydrometeors. [25] An example of this can be seen on the first image on the left in Fig. 7. Rain gauge networks measure accumulated rainfall over time, with a temporal resolution from 1 minute up to 24 hours. Unlike disdrometers that measure the statistical distribution of drop size, rain gauges use point measurements to determine the amount of precipitation. [28]

3.3 Lightning location system

A lightning location system (LLS) is a device used in estimating the location of occurrence of a lightning discharge. Different types of LLS can be used to detect cloud flashes, ground flashes, or total lightning. LLS typically comprises multiple sensors and a central processor. The sensors are responsible for detecting the signals emitted by lightning discharges on certain frequencies and transmitting this information to the central processor.The central processor then uses this data to estimate the location of the discharge, which is ultimately relayed to end-users. LLS with multiple sensors requires a small sensor baseline (the distance between sensors) for efficient lightning detection, as the signal weakens with distance. However, covering the entire Earth with sensors is not practical, so different needs must be fulfilled. [29] LLS can be a network of ground sensors as explained previously or a satellite-based electromagnetic or optical sensor, or a network of sensors.

The frequencies used by sensors can range from ELF to UHF. Lightning location precision refers to how effectively an LLS can detect lightning in comparison to actual lightning events. This performance can be evaluated using various methods and can be determined for specific lightning parameters. The Finnish Meteorological Institute (FMI) is a member of the NORDLIS network, which measures lightning. NORDLIS network is presented in fig. 4. The location accuracy of NORDLIS network is approximately 0.5 - 1 km.

The primary and most commonly used output parameters of an LLS are the spatial and temporal information of a lightning strike. These parameters reveal where and when the lightning occurred. While temporal information based on GPS is highly precise, the spatial accuracy can vary widely depending on both the lightning strike itself and the performance of the LLS. Additionally, the position of the strike in relation to the network geometry also affects the LLS's performance. Lightning data also includes other parameters, such as peak current and multiplicity, which refer to the number of strokes in a ground flash. Finland is using at the time a LLS system called SAFIR 3000 which is controlled by a central processor. The system employs Very high frequency (VHF) interferometry to detect lightning discharges in its coverage area, and it also utilizes a distinct Low frequency (LF) E-field measurement technique to identify ground strokes.The system has the capability to process raw data from both IMPACT and SAFIR sensors. While VHF data is

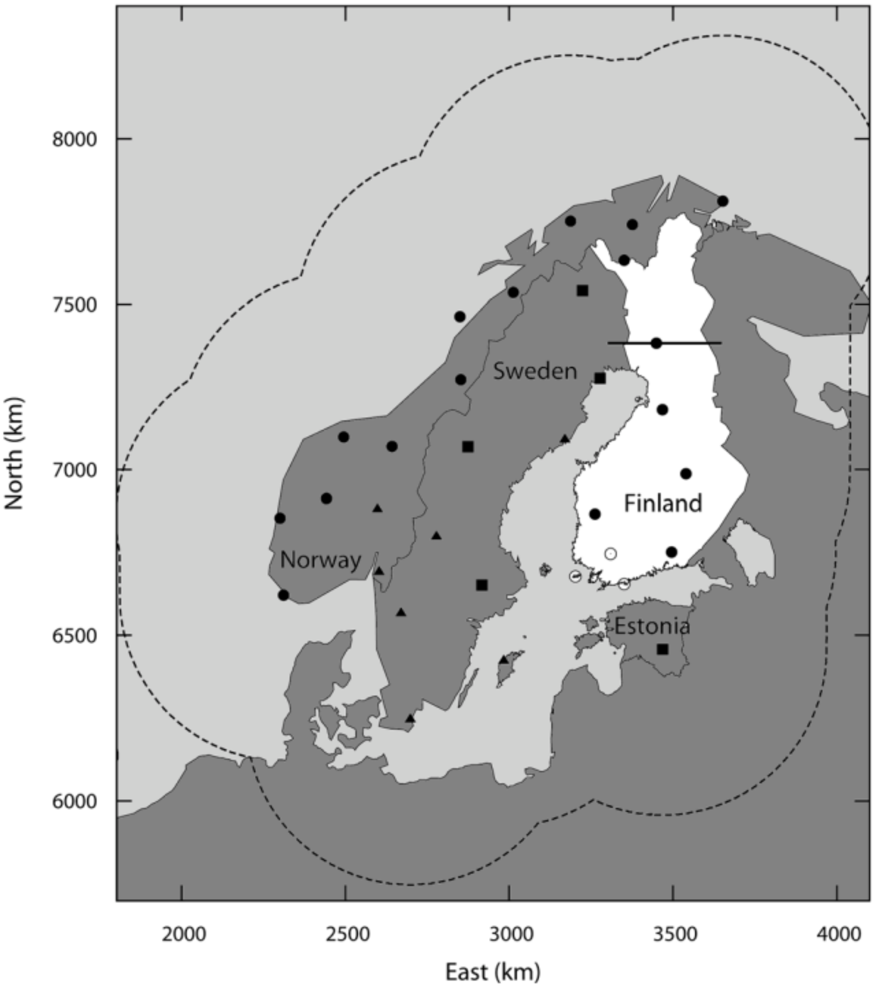

Figure 4. NORDLIS network, the squares, triangles and dots represent different kind of sensors and the dotted area is the coverage area of the network. [Mäkelä, Tuomi, 2010]

processed separately, the system uses the temporal information from the SAFIR sensors' LF data in conjunction with the IMPACT sensors.

The efficiency of LLS can depend on multiple variables. Detection efficiency (DE) is directly affected by both the quantity and the type of sensors within the network along with the networks overall performance. By increasing the sensitivity of sensors and the number of sensors in the network, weaker flashes can be detected, leading to an increase in overall flash count but a decrease in average peak current. However, an excessive number of weak events can raise questions about their nature, whether they are cloud or ground flashes. The number of sensors also greatly impacts the accuracy of the location of detected flashes. Central processor and sensor settings also have a direct effect on the accuracy of results. [30]

The data used in this thesis is based on the LLS sensors and is more accurately presented in Section 4

3.4 Lightning forecasting using weather radar data

Lightning forecasting is a subject of study that focuses on predicting the potential areas affected by thunderstorms. Specific locations of plausible lightning strikes are rather difficult to define, due to the large area of thunder cell and potential strike locations. Two distinct methods are employed for lightning forecasting: the first relies on cloud electrification physics (explicit), while the second employs cloud microphysics and dynamics (diagnostic). The diagnostic approach is computationally more feasible for real-time applications, as it does not entail the use of additional prognostic equations for electrical variables [31, 32]. A recent trend in the field is to combine both lightning and radar data to achieve better forecasts. The main differences in studies are the use of certain weather data, for example, some studies have focused on the effect of heavy rainfall and graupel on lightning strikes, and some focus on the thermodynamical properties of thunderclouds [33, 34]. Mattos et al. [35] have demonstrated a strong correlation between the intrusion of supercooled raindrops into the mixed layer and the VHF source rate density. Specific differential phase (K_{DP}) is radar product that can be used to pinpoint areas of heavy rainfall. Rising K_{DP} suggests a growth in both the dimensions and density of raindrops, resulting in an increase of the rainfall intensity. Additionally, a distinct negative K_{DP} is predominantly observed in higher altitudes in situations with the highest lightning density.

Another study [36] focused on utilizing data from the first C-band polarimetric radar for observing tropical convection, along with a network of Advanced Lightning Direction Finders, a flat plate antenna, and a field mill. It was illustrated that the generation of lightning in tropical island, convection is strongly tied to ice-related processes. While it is not possible to establish a direct causal relationship from remote measurements, a significant correlation between the mass of mixed-phase ice and lightning has been established in a severe tropical island thunderstorm. The relation of intracloud lightning (IC) as an indicator of cloud to ground (CG) lightning has also been studied [37], and it was found that utilizing the first IC flash

as a predictor for the occurrence of CG lightning has been shown to statistically outperform other predictors.
The results of previously examined studies give a premise for the focus of this thesis; Defining the correlation between meaningful weather data variables and lightning strikes and investigating the the coefficient of determination.

3.5 Sensor data

In this section, we analyze the most important variables related to forecasting thunderstorms and lightning strikes. The dBZ (Decibel relative to Z) is a logarithmic, dimensionless technical unit commonly employed in weather radars. It serves the purpose of comparing the equivalent reflectivity factor (Z) of a distant object (expressed in mm^6 per m^3) to the reflection from a rain droplet with a 1 mm diameter (equivalent to 1 mm^6 per m^3) For an exponential distribution of reflector diameters, Z is expressed as:

$$Z = \int_0^{D_{\max}} N_0 e^{-\Lambda D} D^6 dD \tag{6}$$

Where N_0 is number and D_{max} is size of hydrometeors. When dividing the equivalent reflectivity factor (Z) by the reflection from a 1 mm droplet within a cubic meter (Z_0) and then taking the logarithm of the quotient,the logarithmic reflectivity value (L_Z) in dBZ can be obtained. [38].

$$L_Z = 10 \log_{10} \frac{Z}{Z_0}. \tag{7}$$

Values of dBZ can be converted to rainfall rates (R) in millimetres per hour using the Marshall-Palmer formula [39].

$$R = \left(\frac{10^{(dBZ/10)}}{200} \right)^{\frac{5}{8}}. \tag{8}$$

Dual-polarization radars that are used as the source of weather radar data in this thesis can produce information called differential reflectivity (ZDR). ZDR measures reflectivity in both horizontal and vertical directions. The reflected data is converted with a logarithmic ratio that compares the size of the reflectivity return in both directions. The formula for the differential reflectivity is given by:

$$ZDR = 10 \cdot \log_{10}\left(\frac{Z_h}{Z_v}\right), \tag{9}$$

where: ZDR represents the differential reflectivity measured in decibels (dB), Z_h corresponds to the radar reflectivity factor observed in the horizontal polarization and Z_v is the radar reflectivity factor in the vertical polarization. [40]

3.6 Machine learning

Machine learning (ML) has been trending for the past decade, making its way towards natural science and its applications. The main goal of ML algorithms is to imitate the decision-making process of the human brain while gradually improving accuracy. The most common trends in machine learning today, based on the number of publications, are neural networks (Artificial Neural Networks - ANNs). Machine learning is used in physics to replace theoretical methods and improve the calculation speed. Machine learning has also gathered interest in problems that contain multiple variables so that it can be used to replace numerical approximations. It can be described as a set of algorithms, that adapt to changes in the desired way. The idea is that the computer's ability to solve a problem evolves through experience. Such a situation can be created by providing the computer with training data, typically consisting of the initial state, or input, and the desired outcome. The training data is used to create an algorithm that should arrive at a similar result for all inputs that resemble the training data. [41]

3.7 Training of neural networks

Patter and image recognition fall in the category of supervised machine learning. It is defined by the use of features and labeled datasets to train algorithms. The learning process can be divided into three phases; training, testing and validation. During the training phase, the training data serves as input and the learning algorithm extracts features of interest. This constructs the learning model. In the testing phase, the model provides predictions for the test data. The labeled data produced by the learning model represents the final predictions. The sets needed for the process are: training, validation and testing set. The validation set is used after the model has been trained with the initial training set. This helps evaluate the models performance without bias from the training data and is necessary for fine tuning the parameters in the networks architecture. Validation data can also aid in regularization by stopping training when validation error increases, indicating overfitting, a situation where the model corresponds too closely to a particular set of data.

A feature is a specific measurable aspect or trait of the subject of interest. Features are usually in numerical format but graphs and images can also be used [42]. Numeric feature can be described as feature vector, which can be used as input for perceptron, an algorithm used in binary classification. Labeling focuses on identifying and tagging details of interest in data.

Loss function can be used to measure how well the neural network predicts the data. For binary classification problems, the most commonly used loss function is called binary cross-entropy (BCE), also known as softmax loss function. Mathematically, it is defined as:

$$H(y, \hat{y}) = -\frac{1}{N} \sum_{i=1}^{N} (y_i \cdot \log(\hat{y}_i) + (1 - y_i) \cdot \log(1 - \hat{y}_i)) \tag{10}$$

BCE measures the dissimilarity or error between the true binary labels and the predicted values for each example in a dataset. [43]

To increase the size of the training data sets, data augmentation can be used. The main point of data augmentation is to create modified copies of the data that can be used to improve the model. It is useful when the datasets available are small in size. Data augmentation can be used to avoid overfitting issues. For image augmentation geometric transformations are the most straightforward option. The training data can be for example randomly cropped, flipped, zoomed or stretched. [44]

3.8 Binary classification

Binary classification is a machine learning task that involves categorizing elements into one of two predefined groups or classes. This method is applied using a classification rule and is particularly useful for assigning new probabilistic observations to these predefined categories. When the classification task involves only two categories, it is commonly referred to as statistical binary classification.

A decision tree is a classifier used in data analysis, expressed as a recursive partitioning of the instance space. This tree structure comprises nodes forming a rooted tree where there is one node designated as the "root" with no incoming connections. Other nodes in the tree have precisely one incoming connection. Nodes with outgoing edges are termed internal or test nodes, while the rest are referred to as leaves. These leaves either assign a class or provide a probability vector for target values. Instances traverse the tree from root to leaf based on test outcomes. [45]

One of the most commonly used methods in this field is called random forest, also known as random decision forests. [46] A Random Forest consists of a collection of decision trees, with each tree being autonomously built using a distinct subset of the training data. This procedure, known as bootstrapping, introduces randomness that effectively mitigates overfitting, enhancing the model's robustness against data noise.

3.8.1 Convolutional neural network

Convolutional neural network (CNN) is a type of ANN that is typically used to analyze visual imagery. Typical structure of CNN network is presented in Figure 5. CNN consists of input, output, and several hidden layers. Input is the data from

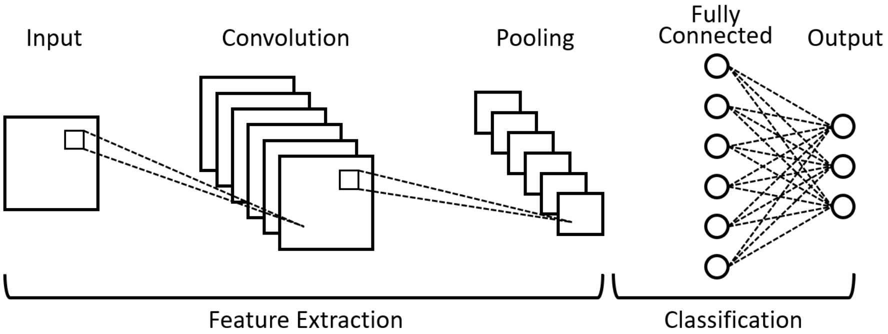

Figure 5. Structure of CNN

which features, and the properties used in prediction are extracted. Generic hidden layer consists convolutional layer, a pooling layer, and a fully connected classifier. Each layer contains nodes, which are called neurons. Each neuron performs a defined calculation which is then passed on to other neurons deeper in the network. To clarify, there can also be other possible configurations. The critical component, the convolutional layer, is responsible for feature extraction. Convolution involves the mathematical operation between the input image and a convolution kernel, which is small matrix of a specific size MxM. Discrete convolution is expressed mathematically:

$$g(x{,}y) = \omega \cdot f(x{,}y) = \sum_{i=-a}^{a} \sum_{j=-b}^{b} \omega(i{,}j) f(x-i, y-j) \tag{11}$$

where: $g(x{,}y)$ is the filtered image, $f(x{,}y)$ is the original image, ω is the filter kernel. Every element of the filter kernel is considered for $a \leq i \leq a$ and $b \leq j \leq b$. The kernel is slid across the input image, computing the dot product between the kernel and the corresponding regions of the input image, each of which matches the size of the kernel $M \times M$. The output is called a feature map, which is then fed to the other layers, which then extract and learn more features of the input image. The pooling layer has the purpose of reducing the size of the convolved feature map to reduce the computational cost of the process. The classifier is a perceptron that processes the features obtained on the previous layers.

Assume that the input image **I** consists of **R** rows, **C** columns and **D** channels. The input can be defined as a three-dimensional function $\mathbf{I}(\mathbf{x}, \mathbf{y}, \mathbf{d})$. The function's domain is defined by $0 \leq x < R, 0 \leq y < C$, and $1 \leq d < D$, which represent the spatial coordinates, and the values **I** at any given (**x**,**y**,**d**) coordinate is the intensity of the d:th color component at spatial coordinates (x,y). The process of acquiring features in convolutional layers of arrays can be expressed as

$$I_f(x{,}y) = b + \sum_{i=-t}^{t} \sum_{j=-t}^{t} \sum_{k=0}^{D-1} W_{ijk} I(x+i, y+j, k), \tag{12}$$

where: I_f is a feature map, W_{ijk} is $w \times w \times D$ convolution kernel for processing D two-dimensional color channels, b is offset and $t = \frac{w-1}{2}$ [47]. Therefore it can be said that are a type of regularized multilayer perceptrons. Typically, multilayer perceptrons refer to fully connected networks, where each neuron in a given layer is connected to all neurons in the subsequent layer.

One of the key factors in the CNN models architecture is the activation function. Activation function is a mathematical function applied to each neurons output in a layer. The purpose of this function is to introduce non-linearity into the network. It has a crucial role in enabling the model to approximate intricate, continuous relationships in the input data. In other words, the activation function determines which pieces of information within the model should be used and which ones should not be active as the network reaches its conclusion. The activation occurs in a process called forward propagation, in which the data is fed through the network. Relatively little preprocessing is required, meaning that the network learns to optimize the kernels through automated learning which makes CNN preferable option to be used in this thesis.

3.8.2 Recurrent Neural Network

Recurrent Neural Network (RNN) algorithm is a specialized artificial neural network designed to learn from repetitive and sequential data, features an internal cyclic structure. The cyclic structure reflects past learning in the current learning situation through the use of weights. Weights represent the connections established between two units within a neural network. To train these units to progress through the network, adjustments need to be made to the weights associated with the unit signals, either increasing or decreasing them. RNNs are suitable options for weather and reflectivity nowcasting since they handle time-sequence data well. Structure of RNN is shown in Fig. 6

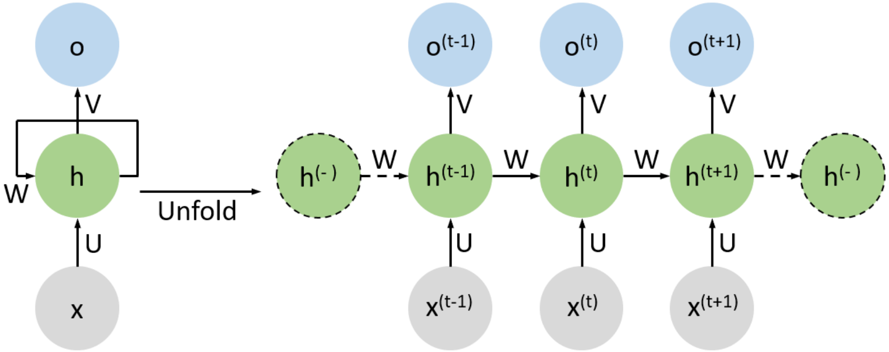

Figure 6. Structure of RNN, x is the input layer, h the hidden layer and o is the output layer

One limitation of RNNs comes from the limited capacity to maintain long-term dependencies. The reason behind this is that the gradient of the loss function decays exponentially with the number of layers through which it is propagated. The backpropagation process involves adjusting the weights of a neural network based on the error rate, also known as loss, obtained in the preceding epoch (iteration). Essentially, during the process, the initial weights can be updated with such small gradients that their values effectively become stagnant.This means that in a multilayer RNN, the weights of the initial layers have very little influence on the final layers over time. As a result, the networks learning ability is degraded. This is known as the vanishing gradient problem. [48]

There has been an increasing amount of published studies about the usage of RNN combined with CNN in complex visual feature recognition problems. [49] Weather prediction models have also been developed using the same methodology [50]. The main challenge with RNN and weather nowcasting is that it has a poor ability to consider future input of the state, due to the vanishing gradient problem so creating future predictions of lightning locations is not directly obtainable with this model.

3.9 Evaluating the models

The models can be evaluated by different metrics. For CNN and image classification, accuracy is the most suitable one. Accuracy is number of correct predictions divided by the number of total predictions. However, accuracy can be biased if the datasets are imbalanced. Imbalanced datasets can occur when the relation between positive and negative classes is skewed. If there are too many negative values compared to the positives, the model will spend most of the time on negative examples and might not learn enough from the positive ones. Precision indicates the quality of a positive prediction made by the model. It is defined as, ie.

$$\text{Precision} = \frac{TP}{TP + FP} \tag{13}$$

where: TP is the number of True Positives and FP is the number of False Positives. Recall represents the fraction of positives that were correctly identified as

$$\text{Recall} = \frac{TP}{TP + FN} \tag{14}$$

F1 combines precision and recall:

$$\text{F1} = 2 \times \frac{\text{Recall} \times \text{Precision}}{\text{Recall} + \text{Precision}} \tag{15}$$

Support is the number of actual occurrences of the class in the specified dataset. [51]

4 Data and methods

The data acquired from the weather radars in Hierarchical Data Format (ODIM HDF5 or h5 in short). The datasets contain multiple different variables with different quantities. The main ones are those presented in Section 4.1; dBZ, ZDR, and the spatial and temporal information about the lightning strike. Lightning data was acquired from archives of the FMI. Multiplicity and peak current were not taken into account, since the main focus of this thesis is to study the ability to predict the location and time of the strike, not the intensity. Datasets were formed from suitable thunderstorm days, in which there was notable lightning strikes recorded. The areas with multiple lightning strikes were cross-referenced with areas that had high reflectivity in terms of dBZ. Depending on the duration of lightning activity, data was gathered for the sets. For example dataset consisted lightning strikes and the relative development of reflectivity as mentioned in section 2.3 from an time interval of 50 minutes in total.

4.1 Preparations of the weather radar and lightning data

Weather radar data used in this thesis is in h5 format. The data contains multiple sweeps on different heights based on the beam angle and multiple variables from which the DBZ and ZDR are extracted. The variables of interest from the lighting location (LL) data are the time of measured discharge and the geographical coordinates of the strike. The distance is limited to 120 km from the measurement point of the weather radar. Distances that are longer than that could contain too much attenuation and radar beam broadening, which would make the data inaccurate for usage in these machine learning applications.

The radar data can not be used directly for multiple of reasons. Attenuation happens when the radar beam comes in contact with a large hail or storm so that the beam does not penetrate properly and scatters. By applying the attenuation correction, one can obtain more accurate reflectivity which is better suitable for ML input. [52] Attenuation can be corrected in multiple ways. Since there was a lack of research and information about attenuation correction methods compared to each other, the simplest one was chosen for the data preprocessing. Another factor is that the data is in polar coordinates by the standard due to the nature of the radar scan. The lightning data is in geographical coordinates, therefore the weather data needs to be adjusted so it is in the same coordinate format as the LL data. As a side note, this process could have been done the other way around as well. The corrections were made using ωradlib and Py-ART Python libraries. [53, 54]. After the attenuation and coordinate corrections, the data is normalized so that the features are transformed to be on a similar scale. Afterwards it can be used as input for the ML models.

After the radar data has been adjusted so that it is compatible with lightning location data, data sets for the ML algorithm can be created. The data arrays are first converted to image arrays so that they can be used as input to the CNN. For this thesis, the goal is to make accurate nowcasts to a 10×10 km area. The lightning data is first converted to a scatter plot, based on the distance to the measurement point (the site of lightning location measurement system). After that the plot is converted to a binary image, in which a single cell contains a 1×1 km area, and if there is a recorded lightning strike inside the cell a value of 1 is given. If there isn't a strike the value is 0. The lightning location image array is used as the validation set. Based on the same area reflectivity and differential reflectivity values are extracted from the weather data sets and suitable plots are created with the same sized grid. As mentioned in the theoretical section, the development of a thunder cell happens primarily 30 minutes before the first discharge. Therefore the weather data of interest is taken in different intervals, from 5 to 30 minutes before the first recorded lightning on the validation set to see the effect. The development of DBZ and ZDR is used in 5-minute time resolution to keep track of the development of the cell. The preprocessing of reflectivity can be seen on the Fig. 7.

The reflectivity and differential reflectivity plots are converted to (256x256) image arrays due to the compact coverage area and quicker computation time. The binary images of lightning locations are also converted to the same size so that they are comparable.

The datasets consisted of 40 minutes of reflectivity development in 5 minute intervals. Most of the lightning occur usually in the mature stage of the thundercloud, and the heaviest rain usually happens after the lightning strikes during the dissipating

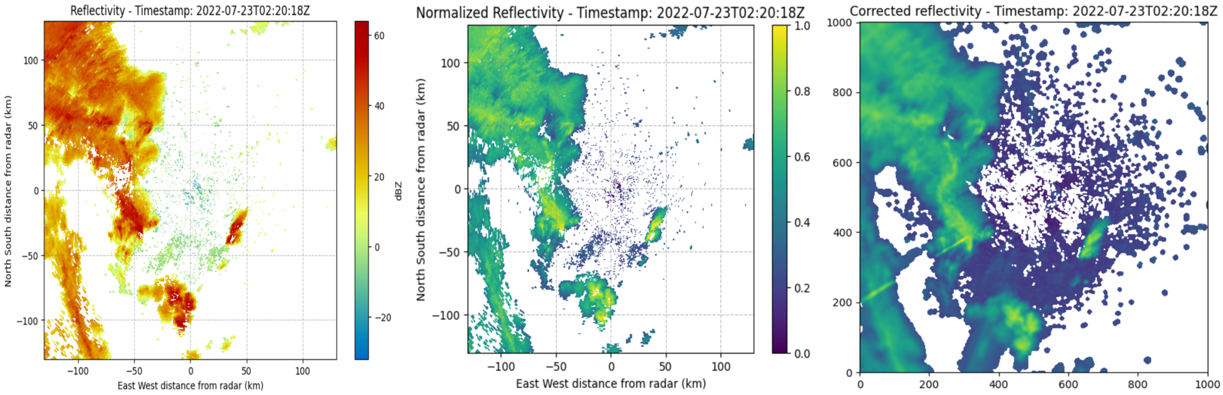
Reflectivity - Timestamp: 2022-07-23T02:20:18Z
North South distance from radar (km)
East West distance from radar (km)
dBZ
Normalized Reflectivity - Timestamp: 2022-07-23T02:20:18Z
North South distance from radar (km)
East West distance from radar (km)
Corrected reflectivity - Timestamp: 2022-07-23T02:20:18Z

Figure 7. Preparing the reflectivity data for model input. Starting with the raw data on the left, normalized in the middle and finally interpolated to Cartesian grid on the right.

stage. The model was tested so that the reflectivity plots corresponded the lightning strike time interval and also so that the reflectivity plots were taken 10 minutes before the first lightning strike.

For the RNN model, the reflectivity data had to be first sequenced to the proper time length. The data set consisted of 60 minutes of lightning activity that was modified to sequences with 5 minute in between. Lightning images were used as labels and reflectivity plots as features. The reflectivity was paired with the corresponding lightning data so that there was a 20-minute gap from the starting time, i.e. the first reflectivity feature image was from time 1:15, and the corresponding label pair was from 01:35 (utm+2). Smaller intervals were also tested but no notable difference could be noticed, so based on theoretical background the 20 minutes was the final choice.

4.2 Machine learning models

The models were constructed using TensorFlow library [55]. The code is represented more accurately in the appendix A.

The CNN model consisted of a 2D convolutional layer of 32 kernels, with kernel size of (3,3) so that the computational cost would be reasonable, rectified linear unit (ReLU) activation function and the input shape of (256, 256, 3) based on the 256x256 pixels and with 3 color channels (RGB) and a pooling window size of (2,2). Then another 2D convolution layer, but with 64 kernels and a duplicate of the pooling layer. Finally a flattening layer and one fully connected layer. The output layer consisted of 1 unit, indicating binary classification, and sigmoid activation function to output probabilities. The output is an image of the predicted lightning strikes based on the reflectivity that occurred in the previous 10 minutes. To create a sequence of predicted images, a combination of CNN and RNN could be suitable.

The model used Adam optimizer and the loss function was binary-cross entropy based. [56] Loss function was selected due to the nature of predicting binary hit or miss labels. The same loss function could be used even if there were more features involved. Taking more classes beyond that would have then required categorical cross entropy function. The model was evaluated using accuracy as a metric.

The RNN model consisted of sequential model with an Long Short-Term memory (LSTM) layer. LSTM is designed to overcome the challenges of learning long-term dependencies in sequence data using structure that includes memory cells and gating mechanisms. [**?**] The layer had 64 units and with expected input sequences of (64,64) and a dense layer with 1 units that served as the output layer. Mean absolute error (MAE) was calculated between the true test labels and the model predictions. The timescale was kept at the 5 minute as explained in Section 4.1. Due to small size of training sets and the lack of features, increasing the timescale would have only decreased the accuracy due to the gradient vanishing.

The RNN model model was a Simple Recurrent Neural Network layer (SimpleRNN) configured with 64 units. This layer was tailored to handle input sequences with a shape of (64, 64), which corresponds to the sequence length and feature dimension of our dataset. After the recurrent layer a Flatten layer was integrated. The inclusion of this layer was to convert the 2D output from the RNN layer into a 1D array. This transformation was necessary to facilitate the transition of data to the subsequent dense layers. A Dense layer was incorporated consisting of 64 neurons, employing a 'relu' activation function. This layer served as an intermediate stage, enhancing the models ability to learn patterns from the flattened data received from the previous layer. Finally a dense layer equipped with a single neuron was applied. This layer utilized a sigmoid activation function. The use of sigmoid activation is a common in binary classification tasks.

5 Results and error analysis

The CNN performed sufficiently compared to the size of used datasets. Reflectivity alone isn't optimal way to predict lightning accurately and more features from weather radar data would be preferable. Data augmentation methods had to be used in order to prevent overfitting from the CNN model. A simple random forest model was also built and trained to show that a less complex solution could be viable when using small data sets. The RNN model performed similarly as the CNN.

5.1 Simple Convolutional Neural network

The results for the CNN are presented in Table 1

Table 1. Classification Report for the CNN model

Class	Precision	Recall	F1-Score	Support
0	0.76	1.00	0.86	134
1	1.0	0.64	0.78	121
Accuracy			0.83	255
Macro Avg		0.88	0.82	255
Weighted Avg		0.87	0.83	255

The model performs much better identifying the absence of lightning (class 0) but struggles much more with correctly identifying actual lightning (class 1) with a recall of 0.64. The training and validation accuracy converge to 0.83, which suggests that the model is not overfitting. Macro average computes the metric independently for each class. With values of 0.82 and 0.88 it suggests that on average, the model performs well on both classes. The weighted average computes the metric for each class but takes the support into account. Precision, recall and F1-score are 0.87, 0.83 and 0.82 respectively. The values are very close with the macro averages, indicating that there is no significant imbalance. Even though the classes seem to be balanced by metrics, the dataset did not contain all possible scenarios for class 0. For example there wasn't situations with high reflectivity and no lighting strikes. This can explain the perfect score for class 1, even though on larger datasets it might not perform so well. Another reason might be the model complexity. The training data might not be diverse enough and the model may memorize the data rather than learning to generalize. The models accuracy plot is represented in 8.

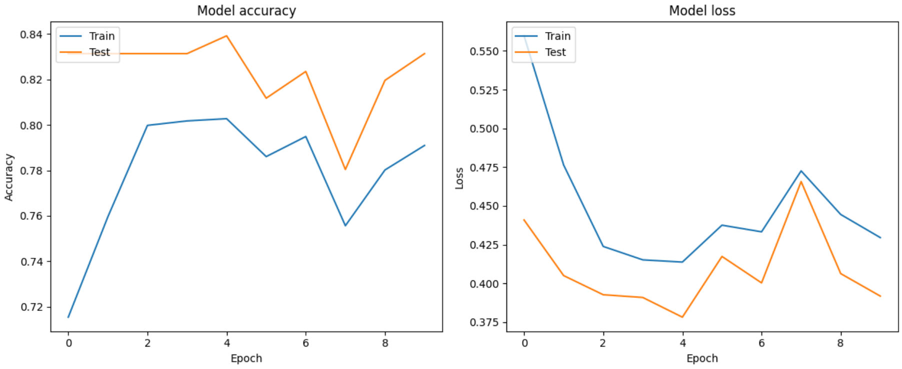

Figure 8. Plots representing the models accuracy and loss over epochs (iterations) for both training and validation data

Overall increasing training accuracy indicates that the model is indeed learning and generalizing. The fluctuations at epoch 7 could be due to several factors. For example the model could be learning noise from the data or there might be variability in the difficulty of the test set samples. The variability in test accuracy and loss suggests that the model could be improved with more data.

5.1.1 Alternative approach for model complexity

Even though the CNN are so called state-of-the-art performance in complex tasks and have been proven on multiple occasions to work with problems such as presented in this thesis, it can be said that more novel approaches can also be made. Especially if the focus is to keep the training samples small in size and the computational cost low. If the goal is the make simple nowcasts on short timescale a random forest model is a suitable solution. The required training time is smaller, results can be more straightforward and the model is less sensitive to noise.

In the Figure 9 is the result visualized from a simple and small run of a random forest model. The confusion matrix is presented in Table 2 and the classification report is presented in Table 3. The confusion matrix compares actual vs. predicted classifications. The matrix was calculated from the test data.

In this case, *True Positive (TP)* = 7058, meaning the correctly predicted positive values. *True Negative (TR)* = 165303, which was just as expected because the

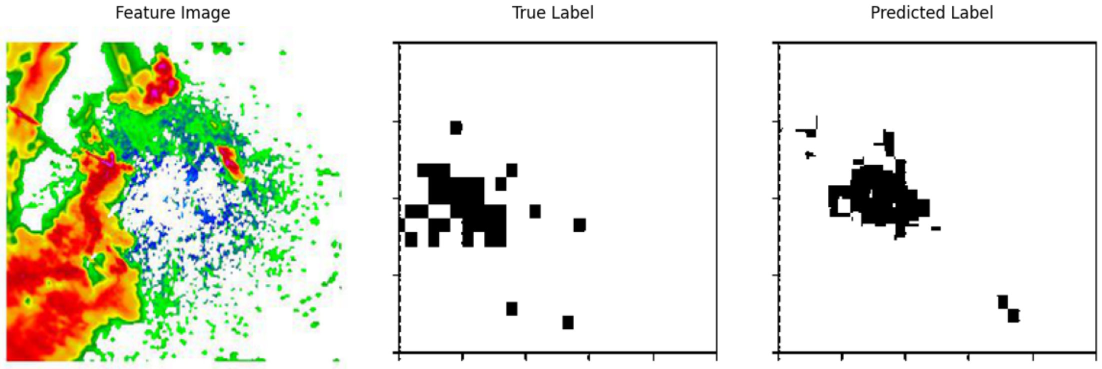

Figure 9. Visualization of the results of random forest model. Normalized reflectivity is on the left, true label on the middle and the models predicted label on the right.

Table 2. Confusion Matrix from test data

		Predicted	
		Class 0	Class 1
Actual	Class 0	7058	17844
	Class 1	6403	165303

statistical area of the lightning strikes in the grid is relatively small. *False Positive (FP)* = 17844 which is defined as actually negative but predicted as positive. Finally *False Negative (FN)* = 6403, actually positive but predicted as negative.

Table 3. Classification Report for the Random Forest model

Class	Precision	Recall	F1-Score	Support
0	0.52	0.27	0.36	24,902
1	0.90	0.96	0.93	171,706
Accuracy			0.88	196,608
Macro Avg		0.71	0.62	196,608
Weighted Avg		0.85	0.88	196,608

Table 3 shows the values for precision which indicates the quality of a positive prediction made by the model, recall is value for how many times the model was able to detect a specific category, F1 combines these both and support is the number of actual occurrences of the class in the specified dataset.

Precision for class 0 is 0.52, and for class 1, it's 0.90. Recall for class 0 is 0.27, and for class 1, it's 0.96. The low result of class 0 in terms of the recall, are due to low volume of training data. The model could also benefit from even smaller scales so that the area in where lightning doesn't strike (0 in the binary images) would be smaller compared to the area of strikes. The results visualized are shown in fig 9

In summary, while the model exhibits proficient precision in identifying instances of class 1, there is a conspicuous deficiency in its ability to accurately detect and predict instances of class 0. That being said, this model was constructed as an example to prove that the concept works and may be more beneficial than the the CNN.

5.2 Recurrent Neural network

The results for the RNN model are presented in table 4. Precision for class 0 is 0.77 and for class 1 its 0.84, meaning that the model is more reliable when predicting if lightning is occurring. With recall value of 0.88 of all actual class 0 instances indicates good sensitivity towards the no lightning events. For class 1 the recall is 0.71 indicating that there is room for improvement. An overall accuracy of 0.80 that is equal with the macro and weighted average indicates a good balance between the classes. The dataset used for this model was smaller than for the random forest, so the results are not directly compatible.

Table 4. Classification Report for the RNN model

Class	Precision	Recall	F1-Score	Support
0	0.77	0.88	0.82	134
1	0.84	0.71	0.77	121
Accuracy			0.80	255
Macro Avg		0.81	0.80	255
Weighted Avg		0.81	0.80	255

5.2.1 Data formatting

To address the obtained results and the lack of sufficient accuracy, it is essential to perform a comprehensive error analysis. The primary factors contributing to the performance limitations are related to attenuation correction, alterations in the coordinate system, and unfavorable dataset size.

The radar data and the used libraries had some unresolved issues with data formatting. Originally the used data was in the h5 format provided by the FMI.

Using wradlib and pyart in the prepossessing did not make any difference in the data, so it was noted that the raw data should be used instead. Curiously, the raw data has same data structure as the .h5 but still works better with the used libraries and for that the proper attenuation calculations can be conducted. The corrections for differential reflectivity ended up blurring the plots, with no notable differences on the plots and as the reason for this lies in the data itself, it needs more researching what library or libraries would be more suitable for this. One one option would be to transfer the data first on some other format, but noting that Vaisala radar data has been used globally with similar models and with viable results, the topic needs more investigating.

Coordinate system transformation was originally planned to be done using the wradlib [53] library. With the unresolved issues with data format and differential reflectivity, approach to use only pyart [54] was made. The pyart contains function that provides gridding to Cartesian coordinates for the radar data. However the results were quite similar as with wradlib. Reflectivity corrections and the plotting works without issues but with differential reflectivity the issue persisted. The issues were checked with the access to the FMI radar archive where the same data was inspected. Pyart has previously worked in similar usage, so the main cause of the issue has yet to be resolved.

5.2.2 Resolution

Lightning location accuracy on 1×1 km grid isn't in any means accurate, since the location data can be achieved much more accurately from the sensors. Reflectivity data on the other hand is difficult to accurately represent in such resolution. For example, if there is 3×3 1 km grid boxes in which half have measured lightning strikes, the reflectivity can still be the same on all the grids due to the resolution. Therefore accurate nowcasting on smaller resolution isn't probably achievable with reflectivity alone. On the other hand if resolution is increased to a larger area by making the grid larger, singular lightning strikes become very difficult to predict due to the large area of high reflectivity.

Both differential reflectivity and reflectivity should nevertheless give reasonable resolution even for grid sizes 5×5 km, so the creating accurate smaller grids would require large amounts of data and/or more features. In hindsight, lightning density map would have been a more suitable label for this model, since it could be done on a larger sized grid. With larger grid the normalization would have resulted in more visible differences.

5.2.3 Data Augmentation

Data augmentation methods were applied to CNN model with modest results. Geometric transformations had no viable effect to the accuracy. This most likely to the previously mentioned resolution difference and hence, making changes to the model geometry had no improving effect. It should be also mentioned that the augmentation methods used were flipping and zooming, and neither one of them had notable positive or negative effect. The goal was to figure out if the lack of training sets was affecting the model, but based on the observed results it was not having a considerable effect.

6 Conclusions and future work

As the results show, the accuracy and selected methods were not suitable for the nowcasting task, at least in the limited form presented in this thesis. First and foremost, one of the main research papers that influenced the topic of this thesis was that research paper of Leinonen and the usage of multiple data sources in the predicting model.[3] Due to this bias, even though the one of topics of this thesis is to find out if the same could be done with as few sources as possible, the model choice was no optimal. In hindsight, a hybrid model combining RNN and CNN would probably been the most suitable.

The usage of reflectivity as the only input feature is not an optimal method for making efficient nowcasting models, with the purpose of getting accurate lightning location data. Reflectivity itself doesn't contain enough information that could explain the electrification locally. For larger areas, the thunder cells and the possible discharging of them could be nowcasted and tracked based on reflectivity alone. This could be possibly examined in the future, the tracking of suitable thunder cells could be combined with radar data to make efficient nowcasts about the lightning strikes. With the models proposed in this thesis, a more suitable choice for the use and prediction of lightning location would be predicting the number of flashes or strikes in a certain area. The area could be 10×10 km or 20×20 km and the task would be to identify such 'hotspots' of discharging, with the highest activity. The relation to reflectivity would be more suitable for this case and the usage of reflectivity data alone could be sufficient. On the other hand, combining other features to the model could also be used, such as the intensity of wind and its trajectory. To take the model even further, geographical data could then be taken into account, if the trajectory could be forecasted in advance.

A more appropriate approach for nowcasting models might involve a shift in perspective, focusing on predicting rain and high reflectivity patterns based on the spatial distribution of lightning strikes. This model would also benefit from the usage of lightning density mapping instead of binary 'hit-miss' -data. The difficulty would probably then be the physical shape of the thunder cell(s) and the possible merging of said cells.

References

[1] A. Mäkelä. Nordic lightning information system: Thunderstorm climate of northern europe for the period 2002–2011. *Atmospheric Research*, 139:46–61, 2014. ISSN 0169-8095. doi: 10.1016/j.atmosres.2014.01.008.

[2] C. Doswell III. Severe convective storms—an overview. *Meteorological Monographs*, 28, 11 2001. doi: 10.1175/0065-9401-28.50.1.

[3] J. Leinonen, U. Hamann, U. Germann, and J. R. Mecikalski. Nowcasting thunderstorm hazards using machine learning: the impact of data sources on performance. *Natural Hazards and Earth System Sciences*, 22(2):577–597, 2022. doi: 10.5194/nhess-22-577-2022.

[4] H. Volland. *Handbook of Atmospheric Electrodynamics*, volume 1. 1995. doi: 10.1201/9780203719503.

[5] R. G. Harrison. Fair weather atmospheric electricity : its origin and applications. 2014. URL `https://api.semanticscholar.org/CorpusID:40682346`.

[6] T. Tuomi. Ten year summary 1977-1986 of atmospheric electricity measured at helsinki-vantaa airport, finland. *Geophysica*, 25, 1989.

[7] L. C. Hale. Middle atmosphere electrical structure, dynamics and coupling. *Advances in Space Research*, 4(4):175–186, 1984. doi: 10.1016/0273-1177(84) 90283-7.

[8] T. Tuomi. *Ukkonen ja salamat*, volume 1. 1992.

[9] V. A Rakov and M. A. Uman. Lightning: Physics and effects. 2004. doi: 10.1256/wea.168/03.

[10] A. Ziv and Z. Levin. Thundercloud electrification: Cloud growth and electrical development. *Journal of Atmospheric Sciences*, 31:1652–1661, 1974. doi: 10.1175/1520-0469(1974)031<1652:TECGAE>2.0.CO;2.

[11] C.P.R. Saunders. A review of thunderstorm electrification processes. *Journal of Applied Meteorology*, 32:642–655, 03 1993. doi: 10.1175/1520-0450(1993) 032<0642:AROTEP>2.0.CO;2.

[12] T. Tuomi. The atmospheric electrode effect over snow. *Journal of Atmospheric and Terrestrial Physics*, 44(9):737–745, 1982. doi: 10.1016/0021-9169(82) 90002-2.

[13] D. R. MacGorman, W. D. Rust, and E. R. Williams. *The electrical nature of storms*. 1998.

[14] M. A. Uman. The peak temperature of lightning. *Journal of Atmospheric and Terrestrial Physics*, 26(1):123–128, 1964. doi: 10.1016/0021-9169(64)90113-8.

[15] V. Cooray and L. Arevalo. Modeling the stepping process of negative lightning stepped leaders. *Atmosphere*, 8, 12 2017. doi: 10.3390/atmos8120245.

[16] A. Nag and V. Rakov. Positive lightning: An overview, new observations, and inferences. *Journal of Geophysical Research (Atmospheres)*, 117, 2012. doi: 10.1029/2012JD017545.

[17] V. Rakov. A review of positive and bipolar lightning discharges. *Bulletin of the American Meteorological Society*, 84, 06 2003. doi: 10.1175/BAMS-84-6-767.

[18] B. Vonnegut and R. Orville. Evidence of lightning associated with the yellowstone park forest fire. 1988.

[19] M. Brook, M. Nakano, P. R. Krehbiel, and T. Takeuti. The electrical structure of the hokuriku winter thunderstorms. *Journal of Geophysical Research*, 87: 1207–1215, 1982.

[20] N. Kitagawa and K. Michimoto. Meteorological and electrical aspects of winter thunderclouds. *Journal of Geophysical Research*, 991, 05 1994. doi: 10.1029/94JD00288.

[21] E. R. Williams, M. E. Weber, and R. E. Orville. The relationship between lightning type and convective state of thunderclouds. *Journal of Geophysical Research*, 94:13213–13220, 1989.

[22] M. Peterson, S. Rudlosky, and D. Zhang. Thunderstorm cloud-type classification from space-based lightning imagers. *Mon Weather Rev*, 148(5):1891–1898, May 2020. doi: 10.1175/mwr-d-19-0365.1.

[23] J. T. Allen, I. M. Giammanco, M. R. Kumjian, J. Punge, Q. Zhang, P. Groenemeijer, M. Kunz, and K. Ortega. Understanding hail in the earth system. *Reviews of Geophysics*, 2020.

[24] N. Li, Y. Zhu, and Z. Wang. A discussion on the applicable condition of rayleigh scattering. *International Journal of Remote Sensing Applications*, 5:62–66, 2015.

[25] M. S. Binetti, C. Campanale, C. Massarelli, and V. F. Uricchio. The use of weather radar data: Possibilities, challenges and advanced applications. *Earth*, 3(1):157–171, 2022. doi: 10.3390/earth3010012.

[26] E. Saltikoff, A. Huuskonen, H. Hohti, J. Koistinen, and H. Järvinen. Quality assurance in the fmi doppler weather radar network. *Boreal Environment Research*, 15:579–594, 01 2010.

[27] Vaisala. Weather radar wrm200. 2021. URL `https://www.vaisala.com/sites/default/files/documents/WRM200-Datasheet-B210698EN.pdf`.

[28] S. Michaelides, V. Levizzani, E. Anagnostou, P. Bauer, T. Kasparis, and J. E. Lane. Precipitation: Measurement, remote sensing, climatology and modeling. *Atmospheric Research*, 94(4):512–533, 2009. doi: doi.org/10.1016/j.atmosres.2009.08.017.

[29] A. Mäkelä. Thunderstorm climatology and lightning location applications in northern europe. 11 2011.

[30] A. Mäkelä, T. Tuomi, and J. Haapalainen. A decade of high-latitude lightning location: Effects of the evolving location network in finland. *Journal of Geophysical Research: Atmospheres*, 115(D21), 2010. doi: 10.1029/2009JD012183.

[31] K. M. G. Rabbani, M. J. Islam, A. O. Fierro, E. R. Mansell, and P. Paul. Lightning forecasting in bangladesh based on the lightning potential index and the electric potential. *Atmospheric Research*, 267:105973, 2022. doi: 10.1016/j.atmosres.2021.105973.

[32] M. Gharaylou, M. M. Farahani, M. Hosseini, and A. Mahmoudian. Numerical study of performance of two lightning prediction methods based on: Lightning potential index (lpi) and electric potential difference (pot) over tehran area. *Journal of Atmospheric and Solar-Terrestrial Physics*, 193:105067, 2019. doi: 10.1016/j.jastp.2019.105067.

[33] J. S. Kastman, P. S. Market, N. I. Fox, A. L. Foscato, and A. R. Lupo. Lightning and rainfall characteristics in elevated vs. surface based convection in the midwest that produce heavy rainfall. *Atmosphere*, 8(2):36, 2017. doi: 10.3390/atmos8020036.

[34] F. Gofa, D. Boucouvala, I. Samos, and P. Louka. Lightning potential forecast evaluation and its correlation with thermodynamic indices. *Environ. Sci. Proc.*, 26(1):109, 2023. doi: 10.3390/environsciproc2023026109. Presented at the 16th International Conference on Meteorology, Climatology and Atmospheric Physics—COMECAP 2023, Athens, Greece, 25–29 September 2023.

[35] Enrique Vieira Mattos, Luiz Machado, Earle Williams, and Rachel Albrecht. Polarimetric radar characteristics of storms with and without lightning activity. *Journal of Geophysical Research Atmospheres*, 121, 11 2016. doi: 10.1002/2016JD025142.

[36] Lawrence D. Carey and Steven A. Rutledge. The relationship between precipitation and lightning in tropical island convection: A c-band polarimetric radar study. *Monthly Weather Review*, 128(8):2687 – 2710, 2000. doi: https://doi.org/10.1175/1520-0493(2000)128<2687:TRBPAL>2.0.CO;2.

[37] G. N. Seroka, R. E. Orville, and C. Schumacher. Radar nowcasting of total lightning over the kennedy space center. *Weather and Forecasting*, 27(1):189 – 204, 2012. doi: 10.1175/WAF-D-11-00035.1.

[38] M. K. Yau and R. Rogers. *A Short Course in Cloud Physics*. Elsevier Science, 3rd edition, 1996. URL `https://www.perlego.com/book/1836119/a-short-course-in-cloud-physics-pdf`.

[39] J. S. Marshall and W. M. K. Palmer. The distribution of raindrops with size. *Journal of Meteorology*, 5:165–166, 1948. doi: 10.1175/1520-0469(1948).

[40] T. A. Seliga and V. N. Bringi. Potential use of radar differential reflectivity measurements at orthogonal polarizations for measuring precipitation. *Journal Name*, Volume Number:Page Numbers, 2023. URL `https://api.semanticscholar.org/CorpusID:53578403`.

[41] C. Janiesch, P. Zschech, and K. Heinrich. Machine learning and deep learning. *Electron Markets 31 685–695*, 2021. doi: 10.1007/s12525-021-00475-2.

[42] C. Bishop. *Pattern Recognition and Machine Learning*. Springer, 2006. ISBN 0-387-31073-8.

[43] R. Wali. Xtreme margin: A tunable loss function for binary classification problems. 2022. doi: 10.48550/arXiv.2211.00176.

[44] L. Perez and J. Wang. The effectiveness of data augmentation in image classification using deep learning. *arXiv*, 2017. doi: 10.48550/arXiv.1712.04621.

[45] L. Rokach and O. Maimon. Decision trees. In O. Maimon and L. Rokach, editors, *Data Mining and Knowledge Discovery Handbook*. Springer, 2005. doi: 10.1007/0-387-25465-X_9.

[46] G. Biau and E. Scornet. A random forest guided tour. *TEST*, 25:197–227, 2016. doi: 10.1007/s11749-016-0481-7.

[47] M. Valueva, N. Nagornov, P. Lyakhov, G. V. Valuev, and N. I. Chervyakov. Application of the residue number system to reduce hardware costs of the convolutional neural network implementation. *Mathematics and Computers in Simulation*, 177, 05 2020. doi: 10.1016/j.matcom.2020.04.031.

[48] S. Hochreiter. Recurrent neural net learning and vanishing gradient. *International Journal of Uncertainty, Fuzziness and Knowledge-Based Systems*, 6(2): 107–116, 1998.

[49] S. Gheisari, S. Shariflou, and J. et al. Phu. A combined convolutional and recurrent neural network for enhanced glaucoma detection. *Scientific Reports*, 11(1):1945, 2021. doi: 10.1038/s41598-021-81554-4.

[50] B. Zhao, W. Liu, X. Lu, and Z. Wang. A cnn–rnn architecture for multi-label weather recognition. *Neurocomputing*, 322, 2018. doi: 10.1016/j.neucom.2018.09.048.

[51] Ž. Đ. Vujović. Classification model evaluation metrics. *International Journal of Advanced Computer Science and Applications*, 12, 2021. doi: 10.14569/IJACSA.2021.0120670.

[52] S-G. Park, V. N. Bringi, V. Chandrasekar, M. Maki, and K. Iwanami. Correction of radar reflectivity and differential reflectivity for rain attenuation at x band. part i: Theoretical and empirical basis. *Journal of Atmospheric and Oceanic Technology*, 22(11):1621 – 1632, 2005. doi: 10.1175/JTECH1803.1.

[53] M. Heistermann, S. Jacobi, and T. Pfaff. Technical note: An open source library for processing weather radar data (wradlib). *Hydrol. Earth Syst. Sci.*, 17:863–871, 2013. doi: 10.5194/hess-17-863-2013.

[54] J. J. Helmus and S. M. Collis. The Python ARM Radar Toolkit (Py-ART), a library for working with weather radar data in the python programming language. *Journal of Open Research Software*, 4(1):e25, 2016. doi: 10.5334/jors.119.

[55] M. Abadi, A. Agarwal, P. Barham, E. Brevdo, Z. Chen, C. Citro, G. S. Corrado, A. Davis, J. Dean, M. Devin, S. Ghemawat, I. Goodfellow, A. Harp, G. Irving, M. Isard, Y. Jia, R. Jozefowicz, L. Kaiser, M. Kudlur, J. Levenberg, D. Mané, R. Monga, S. Moore, D. Murray, C. Olah, M. Schuster, J. Shlens, B. Steiner, I. Sutskever, K. Talwar, P. Tucker, V. Vanhoucke, V. Vasudevan, F. Viégas, O. Vinyals, P. Warden, M. Wattenberg, M. Wicke, Y. Yu, and X. Zheng. Tensorflow: Large-scale machine learning on heterogeneous systems, 2015. URL `https://www.tensorflow.org/`. Software available from tensorflow.org.

[56] R. O. Ogundokun, R. Maskeliunas, S. Misra, and R. Damaševičius. Improved cnn based on batch normalization and adam optimizer. In *Computational Science and Its Applications – ICCSA 2022 Workshops*, volume 13381, 2022. doi: 10.1007/978-3-031-10548-7_43.

A Code

A.1 Data preprocessing

A.1.1 Lightning data processing

The following Python code demonstrates how to process the lightning data:

```python
# Initialize empty lists to store extracted values
flash_ids = []
peak_currents = []
latitudes = []
longitudes = []
utctimes = []

# Iterate over the list of dictionaries to extract the required information
for data in results:
    flash_ids.append(data[’flash_id’])
    peak_currents.append(data[’peak_current’])
    latitudes.append(data[’latitude’])
    longitudes.append(data[’longitude’])
    utctimes.append(data[’utctime’])

# Define the boundaries of the 1x1 km grid
min_lon, max_lon = min(longitudes), max(longitudes)
min_lat, max_lat = min(latitudes), max(latitudes)
# 1 degree of longitude is approximately 111.32 km
lon_step = 10.0 / 111.32
lat_step = 10.0 / 111.32

# Create a grid of lon and lat values
grid_lon = np.arange(min_lon, max_lon, lon_step)
grid_lat = np.arange(min_lat, max_lat, lat_step)
```

```

# Create an empty binary grid (initialized with zeros)
binary_grid = np.zeros((len(grid_lat), len(grid_lon)), dtype=int)

# Populate the binary grid based on lightning data points
for lon, lat in zip(longitudes, latitudes):
    if min_lon <= lon <= max_lon and min_lat <= lat <= max_lat:
        col_idx = int((lon - min_lon) // lon_step)
        row_idx = int((lat - min_lat) // lat_step)
        binary_grid[row_idx, col_idx] = 1

# Create a binary grid plot
plt.figure(figsize=(10, 6))
plt.imshow(binary_grid, cmap='binary', extent=[min_lon, max_lon, min_lat, max_lat], origin='lower')

# Customize the plot as needed
plt.grid(True, linestyle='--', color='gray', alpha=0.5)
plt.tight_layout()
plt.ylim()
plt.xlim()
# Remove tick labels while keeping the grid lines
plt.xticks(ticks=plt.xticks()[0], labels=[])
plt.yticks(ticks=plt.yticks()[0], labels=[])

# Show the plot
plt.show()
```

A.1.2 Radar data processing

The following Python code illustrates the processing of radar data:

```python
import numpy as np
import pyart
import wradlib
import matplotlib.pyplot as plt
from osgeo import gdal
from osgeo import osr

file_names = []

for fname in file_names:
    # Load radar data from the current file
    radar = pyart.io.read(fname, include_datasets=["dataset1", "dataset2", "dataset3", "dataset4", "dataset5", "dataset6"])

    # Extracting correct values from data
    z_values = radar.fields[’reflectivity’][’data’]
    zdr_values = radar.fields[’differential_reflectivity’][’data’]
    timestamp = radar.time[’units’][14:]

    latitude = radar.latitude[’data’][0]
    longitude = radar.longitude[’data’][0]
    altitude = radar.altitude[’data’][0]
    sitecoords = (27.38147004, 62.86260009, 269) #example
    sitecoords = list(sitecoords)

    # Defining proper frequency used by the radar
    frequency = 5.33
    radar.instrument_parameters = {"frequency": {"data": np.array([frequency * 1e9])}}

    # Perform attenuation correction with modified parameters
    cor_z = pyart.correct.calculate_attenuation_zphi(
      radar,
      fzl=4000,
      smooth_window_len=1000,
      c=frequency,
      refl_field='corrected_reflectivity',
      phidp_field='differential_phase',
      zdr_field='differential_reflectivity',
      temp_ref='fixed_fzl'
    )

    cor_z_data = cor_z[2]
    cor_zdr_data = cor_z[5]
    corzdr = cor_zdr_data['data']
    corref = cor_z_data['data']

    # Remove existing fields
    radar.fields.pop('corrected_reflectivity')
    radar.fields.pop('corrected_differential_reflectivity')

    # Create grid coordinates (x, y, z) using wradlib's spherical_to_xyz function
    azimuths = radar.azimuth['data']
    elevations = radar.elevation['data']
    ranges = cor_z_values
    ranges_zdr = cor_zdr_values

    coords_z_data = wradlib.georef.spherical_to_xyz(ranges, azimuths, elevations, sitecoords)
    coords_zdr_data = wradlib.georef.spherical_to_xyz(ranges_zdr, azimuths, elevations, sitecoords)

    coords_z = coords_z_data[0]
    coords_zdr = coords_zdr_data[0]

    field_dict = {'data': corref,
         'units': 'dBZ',  # Replace with the actual unit
```

```
           'long_name': 'Corrected Reflectivity',
           'standard_name': 'corrected_reflectivity',
           'comments': 'Reflectivity corrected for attenuation'}

 field_dict = {'data': corzdr,
           'units': 'dB',  # Replace with the actual unit
           'long_name': 'Corrected Differential Reflectivity',
           'standard_name': 'corrected_differential_reflectivity',
           'comments': 'ZDR corrected for attenuation'}

 # Add the corrected reflectivity field to the radar object
 radar.add_field('corrected_reflectivity', {'data': coords_z})
 radar.add_field('corrected_differential_reflectivity', {'data': coords_zdr})

 # Add the corrected reflectivity data to the radar object
 radar.add_field('corrected_reflectivity', field_dict, replace_existing=True)

 gatefilter = pyart.filters.GateFilter(radar)
 gatefilter.exclude_transition()
 gatefilter.exclude_masked("corrected_reflectivity")

 # Create a grid using pyart.map.grid_from_radars()
 grid = pyart.map.grid_from_radars(
     radar,
     grid_shape=(1, 1001, 1001),
     grid_limits=((0, 3700), (-120000.0, 120000.0), (-120000.0, 120000.0)),
     fields=['corrected_reflectivity']  # If 'x_coordinate' is the field you added and want to grid
```

```
    )

    # Plot the interpolated data
    fig = plt.figure(figsize=(8, 6))
    ax = fig.add_subplot(111)
    ax.imshow(grid.fields["corrected_reflectivity"]["data"][0], origin="lower")
    ax.set_title(f'Corrected reflectivity - Timestamp: {timestamp}')
    plt.show()
```

A.2 CNN Model

```
# Import necessary libraries
import cv2
import numpy as np
import tensorflow as tf
from sklearn.model_selection import train_test_split
from tensorflow.keras.models import Sequential
from tensorflow.keras.layers import Conv2D, MaxPooling2D, UpSampling2D, Flatten, Dense

#Define file paths for reflectivity images and lightning strike images
file_paths = []
lightning_paths = []# Initialize lists for storing image data and labels
ref_images = []
light_images = []
ref_labels = []
light_labels = []# Load and preprocess reflectivity images

for path in file_paths:
    img = cv2.imread(path, cv2.IMREAD_GRAYSCALE)
    resized_img = cv2.resize(img, (256, 256))
    ref_images.append(resized_img)
    ref_labels.append(0)  # Label for non-lightning images# Load and preprocess lightning images

for path in lightning_paths:
    img = cv2.imread(path, cv2.IMREAD_GRAYSCALE)
    resized_img = cv2.resize(img, (256, 256))
    light_images.append(resized_img)
    light_labels.append(1)  # Label for lightning images# Combine and normalize image data

all_images = np.array(ref_images + light_images).reshape(-1, 256, 256, 1) / 255.0
all_labels = np.array(ref_labels + light_labels)# Function to extract image patches
def extract_patches(image, patch_size, stride):
    H, W, C = image.shape
    return np.array([image[i:i+patch_size, j:j+patch_size]
                     for i in range(0, H-patch_size+1, stride)
                     for j in range(0, W-patch_size+1, stride)])# Define patch size and stride for patch extraction
patch_size = 64
stride = 32# Extract patches and corresponding labels
all_patches = [extract_patches(img, patch_size, stride) for img in all_images]
all_patch_labels = np.repeat(all_labels, len(all_patches[0])) # Flatten the list of patches
all_patches = np.vstack(all_patches)# Split data into training and test sets
X_train, X_test, y_train, y_test = train_test_split(
    all_patches, all_patch_labels, test_size=0.2, random_state=42)# Define the CNN model architecture
model = Sequential([
    Conv2D(64, (3,3), activation='relu', padding='same', input_shape=(patch_size, patch_size, 1)),
    MaxPooling2D((2,2), padding='same'),
```

```python
    Conv2D(128, (3,3), activation='relu', padding='same'),
    MaxPooling2D((2,2), padding='same'),
    Conv2D(128, (3,3), activation='relu', padding='same'),
    UpSampling2D((2,2)),
    Conv2D(64, (3,3), activation='relu', padding='same'),
    UpSampling2D((2,2)),
    Conv2D(1, (3,3), activation='sigmoid', padding='same'),
    Flatten(),
    Dense(128, activation='relu'),
    Dense(1, activation='sigmoid')
])# Compile the model with optimizer, loss function, and metrics
model.compile(optimizer='adam',
              loss=tf.keras.losses.BinaryCrossentropy(),
              metrics=['accuracy'])# Define training callbacks
callbacks = [
    tf.keras.callbacks.EarlyStopping(patience=5, monitor='val_loss'),
    tf.keras.callbacks.TensorBoard(log_dir='./logs')
]# Train the model with the training data
history = model.fit(
    X_train, y_train,
    epochs=10,
    validation_data=(X_test, y_test),
    batch_size=2,
    callbacks=callbacks
)
```

A.3 RNN model

```
#RNN model architecture
model = Sequential([
    SimpleRNN(64, return_sequences=False, input_shape=(64, 64)),
    Flatten(),
    Dense(64, activation=’relu’),
    Dense(1, activation=’sigmoid’)
])# Compile the model
model.compile(optimizer=’adam’,
              loss=tf.keras.losses.BinaryCrossentropy(),
              metrics=[’accuracy’])# Train the model
history = model.fit(
    X_train.reshape(-1, 64, 64), y_train,
    epochs=10,
    validation_data=(X_test.reshape(-1, 64, 64), y_test),
    batch_size=2
)
```