Feature extraction for classification of auroral images

Shwetha Herga
FEATURE EXTRACTION FOR CLASSIFICATION OF AURORAL IMAGES

by

SHWETHA HERGA

A THESIS

Submitted in partial fulfillment of the requirements for the degree of Master of Science in The Department of Computer Science to The School of Graduate Studies of The University of Alabama in Huntsville

HUNTSVILLE, ALABAMA

2020
In presenting this thesis in partial fulfillment of the requirements for a master’s degree from The University of Alabama in Huntsville, I agree that the Library of this University shall make it freely available for inspection. I further agree that permission for extensive copying for scholarly purposes may be granted by my advisor or, in his/her absence, by the Chair of the Department or the Dean of the School of Graduate Studies. It is also understood that due recognition shall be given to me and to The University of Alabama in Huntsville in any scholarly use which may be made of any material in this thesis.

Shwetha Herga

06/22/2020 (date)
THESIS APPROVAL FORM

Submitted by Shwetha Herga in partial fulfillment of the requirements for the degree of Master of Science in Computer Science and accepted on behalf of the Faculty of the School of Graduate Studies by the thesis committee.

We, the undersigned members of the Graduate Faculty of The University of Alabama in Huntsville, certify that we have advised and/or supervised the candidate of the work described in this thesis. We further certify that we have reviewed the thesis manuscript and approve it in partial fulfillment of the requirements for the degree of Master of Science in Computer Science.

**Huaming Zhang**

Dr. Huaming Zhang (Date) Committee Chair

**Dr. Ying Zou**

06/24/2020

Dr. Ying Zou (Date)

**Dr. Tathagata Mukherjee**

06/23/2020

Dr. Tathagata Mukherjee (Date)

**Department Chair**

Dr. Heggere S. Ranganath (Date) Department Chair

**Interim Dean, College of Science**

Dr. John Christy (Date)

**Graduate Dean**

David Berkowitz

Dr. David Berkowitz (Date)
Auroras are a dynamically evolving phenomenon. Different auroral forms are correlated with various physical processes in the magnetosphere and ionosphere system. Millions of auroral images are captured every year by the modern ground-based All-Sky Imager (ASI). In dealing with data from ASI, machine learning techniques play a critical scientific role, facilitating both efficient searches and statistical studies. In this work, we manually label night-side auroral images from various Time History of Events and Macroscale Interactions during Substorms (THEMIS) all-sky imager based on the sky conditions; the labels are clear sky with auroras, cloudy with the moon, cloudy, clear-sky with the moon, and clear-sky. This is an interdisciplinary work between space science and computer science disciplines. A deep convolutional neural network is developed with auroral images as input for training. The deep learning model is trained to classify the images into five classes based on the extracted features. The central aspect we are concerned with is what the network learns about auroral features and how it learns as it convolves into deeper iterations. In comparison with conventional techniques, the proposed model achieves a high classification accuracy.
ACKNOWLEDGMENTS

I would like to express my sincere gratitude to my thesis advisor, Dr. Huaming Zhang, for his constant support and guidance throughout my Master’s program. The door to Dr. Zhang’s office was always open whenever I ran into research problems or had questions about my research or writing. He consistently allowed this thesis to be my work but steered me in the right direction whenever he thought I needed it.

I express profound gratitude to Dr. Ranganath Heggere, Professor and Chair of the Department of Computer Science, for having provided me with all the facilities to carry out this research work.

I would like to acknowledge Dr. Ying Zou for her valuable input, constant support, and guidance to learn new concepts about space physics and Dr. Tathagata Mukherjee as committee members of my thesis defense. I am gratefully indebted to their valuable comments on this thesis.

A special thanks to my husband Deepak Bhaskar Acharya and my dear friend Dr. Swetha Govindaiah for their constant encouragement and guidance.

Finally, I must express my very profound gratitude to my father Mr. Damodar Acharya, mother Mrs. Rekha Damodar, sister Shruthi Herga, father-in-law Mr. Bhaskar Acharya, mother-in-law Mrs. Shobha Bhaskar Acharya for providing me with unfailing support and continuous encouragement throughout my years of study and through the process of researching and writing this thesis. This accomplishment would not have been possible without them.
# TABLE OF CONTENTS

**List of Figures** 10

**List of Tables** 12

1 Introduction 1
   1.1 Motivation .................................................. 1
   1.2 Problem Statement ........................................... 2
   1.3 Proposed Method ............................................. 3
   1.4 Thesis Outline ............................................... 4

2 Background and Related Work 6
   2.1 Overview ..................................................... 6
   2.2 Classification with CNN ....................................... 7
   2.3 Current Advancements ......................................... 10
   2.4 Summary ..................................................... 11

3 Methodology 12
   3.1 Theoretical Overview of CNN ................................. 12
      3.1.1 Convolutional Neural Network ............................ 12
      3.1.2 Max Pooling .............................................. 14
      3.1.3 Reduce Overfitting ...................................... 16
3.1.4 Batch Normalization ............................................. 17
3.1.5 Dropout ............................................................. 17
3.1.6 ReLU Non-Linearity .............................................. 18
3.1.7 Softmax layer and Loss function .............................. 18
3.2 CNN based Classification Methodology ......................... 20
   3.2.1 Proposed Method ............................................... 20
   3.2.2 Methodology for Comparison .................................. 24
      3.2.2.1 Proposed Architecture .................................. 24
      3.2.2.2 Alexnet .................................................... 26
      3.2.2.3 VGG ...................................................... 28
      3.2.2.4 Resnet .................................................... 29
3.3 Experimental Design ............................................... 31
4 Experimental Evaluation ............................................. 32
   4.1 Data Preprocessing ............................................... 32
      4.1.1 Instrumentation and Data .................................. 32
      4.1.2 Data Pre-Processing ....................................... 34
      4.1.3 SpacePy ..................................................... 34
   4.2 Training and Testing ............................................. 38
   4.3 Evaluation Results ............................................... 39
      4.3.1 Confusion Matrix .......................................... 41
   4.4 Feature Extraction of Auroral Images ......................... 42
   4.5 Evaluation of activation functions ............................ 46
# LIST OF FIGURES

<table>
<thead>
<tr>
<th>FIGURE</th>
<th>PAGE</th>
</tr>
</thead>
<tbody>
<tr>
<td>3.1 Example of Convolutional Neural Network [1]</td>
<td>14</td>
</tr>
<tr>
<td>3.2 Overlapping Pooling [10]</td>
<td>15</td>
</tr>
<tr>
<td>3.3 Overlapping Max Pooling</td>
<td>16</td>
</tr>
<tr>
<td>3.4 Sample Images of each Categories. (a) Clear sky with aurora (b) Cloudy with the moon (c) Cloudy (d) Clearsky with the moon (e) Clearsky</td>
<td>23</td>
</tr>
<tr>
<td>3.5 Proposed Architecture</td>
<td>26</td>
</tr>
<tr>
<td>3.6 Alexnet Architecture [2]</td>
<td>27</td>
</tr>
<tr>
<td>3.7 VGG Architecture [3]</td>
<td>29</td>
</tr>
<tr>
<td>3.8 Resnet Architecture [4]</td>
<td>31</td>
</tr>
<tr>
<td>4.1 Map of North America with GBO Station Names, Fields of View of the All Sky Imagers with Locations. [5]</td>
<td>33</td>
</tr>
<tr>
<td>4.2 Confusion Matrix</td>
<td>42</td>
</tr>
<tr>
<td>4.3 Input Image to the Convolutional Neural Network</td>
<td>43</td>
</tr>
<tr>
<td>4.4 Feature map of Convolutional Layer-1</td>
<td>44</td>
</tr>
<tr>
<td>4.5 Feature map of Convolutional Layer-2</td>
<td>44</td>
</tr>
<tr>
<td>4.6 Feature map of Convolutional Layer-3</td>
<td>45</td>
</tr>
<tr>
<td>4.7 Feature map of Convolutional Layer-4</td>
<td>45</td>
</tr>
<tr>
<td>4.8 Feature map of Convolutional Layer-5</td>
<td>46</td>
</tr>
<tr>
<td>4.9 Loss Comparison of ReLU vs Sigmoid</td>
<td>47</td>
</tr>
</tbody>
</table>
4.10 Accuracy Comparison of ReLU vs Sigmoid ...

4.11 Accuracy Comparison plot ...
# LIST OF TABLES

<table>
<thead>
<tr>
<th>TABLE</th>
<th>PAGE</th>
</tr>
</thead>
<tbody>
<tr>
<td>3.1</td>
<td>Description of Classes with Corresponding Labels</td>
</tr>
<tr>
<td>3.2</td>
<td>Architecture of our Network</td>
</tr>
<tr>
<td>4.1</td>
<td>Number of Images Per Year Based on the Sky Conditions</td>
</tr>
<tr>
<td>4.2</td>
<td>Count of the Images Used in our Experiment from the ASI Stations</td>
</tr>
<tr>
<td>4.3</td>
<td>Count of the Images Used in our Experiment from the ASI Stations</td>
</tr>
<tr>
<td>4.4</td>
<td>Count of the Images Used in our Experiment from the ASI Stations</td>
</tr>
<tr>
<td>4.5</td>
<td>Count of the Images Used in our Experiment from the ASI Stations</td>
</tr>
<tr>
<td>4.6</td>
<td>Hyperparameters of the Proposed Architecture Used for Classification</td>
</tr>
<tr>
<td>4.7</td>
<td>Precision, Recall and F1 score for Different Classifiers on Auroral Dataset</td>
</tr>
</tbody>
</table>
CHAPTER 1

INTRODUCTION

1.1 Motivation

Auroras are the beautiful shimmering display of light formed due to charged particles (mostly electrons but also protons) from near-earth space, accelerated along magnetic field lines, and eventually colliding with the neutral constituents of the upper atmosphere (primarily atomic oxygen). Since the enormous space of the magnetosphere connects with the upper atmosphere through magnetic field lines, the atmosphere serves as a screen where magnetosphere dynamics get projected [6]. The observation of aurora from the ground allows one to observe large-scale magnetospheric processes both on the day but also on the nightside.

Millions of auroral images have been taken in the Arctic and Antarctic regions, since the first auroral imaging campaigns. Auroral scientists now have access to more data than is humanly possible to handle by visual inspections. Machine learning methods have shown to be a valuable and highly applicable tool for the classification of large image datasets over recent decades. The study of the classification of aurora thus assists in the interpretation of each auroral type’s physical structure and is of considerable significance in the aurora research.
Computer vision is a branch of computer science where techniques are studied and developed for automatic image analysis and processing. Machine learning approaches are a powerful and widely applicable tool for classification (supervised learning), and clustering (unsupervised learning) \[7\] large datasets samples. Automatic classification of images captured each year will make it easier for scientists to study the images of interest in an organized, objective, and increase efficiency and effectiveness \[8\]. Automating the classification process helps the machine learning model to look through large sets of images, collect information, and learn patterns and identify them.

Auroral classification involves the extraction of the features and the classifier’s design. The design of features by hand is often a long process that requires expertise to design according to specific data and tasks delicately \[9\]. Recently, deep learning algorithms, due to their discriminatory capability, have dominated most vision-based tasks. Image representations based on convolutional neural network (CNN) have gained more attention, and demonstrates the excellent performance \[10\] \[11\] \[12\]. The major difference between CNNs and conventional machine learning methods is that CNNs learn image features directly without additional manual feature extraction.

1.2 Problem Statement

The purpose of NASA Time History of Events and Macroscale Interactions during Substorms (THEMIS) project is to research intended to investigate the magnetospheric substorm, which are manifestations of the instability of the magnetosphere, and an explosive release of solar wind energy stored within the earth’s magnetotail.
The techniques for classifying the auroral images include two steps of procedures: feature extraction and the classifier’s design. The drawn characteristics reflect the intensity, shape, and texture of the sky conditions and play an essential role in the classification of auroral images.

The primary objective of the thesis is to provide an efficient means for the extraction of features present in the auroral images for useful auroral classification. In this regard, we are mainly interested in what the network parameters learn about auroral features, and how it knows as it transitions to deeper model iterations. Based on the extracted features, the trained machine learning model can correctly predict the unseen auroral images.

1.3 Proposed Method

Convolutional neural networks are unique sorts of deep neural networks (DNNs) that utilize discrete convolutions to process large datasets and concentrate highlights. These networks are particularly helpful with regards to image order, due to the convolutional layers selecting abstract features independent of their image. Recent studies have proven that CNN can automatically learn the intricate pattern and characteristic features from the data in the hierarchical stream.

In the field of machine learning, the extraction of features has gained growing attention. The urge to provide informative features is crucial for detecting patterns, extracting information, or predicting future observations from big data. The algorithm falls into the supervised learning category that we use to classify auroral images using
labeled data. We train our model in supervised learning, and once the model is optimized, predictions of unseen or future data become easy.

In this study, the CNN approach automatically learns the network with little human intervention from the auroral images and gives higher precision than established procedures. The CNN layers are visualized and analyzed. In our research, we evaluate and analyze five activation functions and four pooling layers operations, which are crucial components of CNN, focusing on their effect on the training of the machine learning model efficiency and classification accuracy of auroral classification. This research is interdisciplinary and involves a knowledge of the work of computer science and space physics. The network automatically identifies the aurora into five different classes: clear sky with auroras, cloudy with the moon, cloudy, clear sky with moon, and clear sky based on the sky conditions. The essential factor is to evaluate features learned with a little human interference by the convolutional neural network and to provide greater accuracy with existing approaches.

1.4 Thesis Outline

The thesis’s remaining section is sorted out as follows: In Chapter 2, a comprehensive introduction to the aurora images with a brief foreword to the acquisition of auroral data, followed by a detailed discussion on the application of auroral images. In Chapter 3, we discuss the theoretical description and hypothesis behind all the components of the proposed architecture presented along with their mathematical representations. This is accompanied by the discussion of the proposed feature extraction architecture and other machine learning models that are used to compare
the experimental evaluation of the proposed techniques. In Chapter 4, we discuss the
deep learning model-based feature extraction and validation through experimental
results. Finally, in Chapter 5, we discuss the summary of the presented work and the
avenues for future research.
CHAPTER 2

BACKGROUND AND RELATED WORK

2.1 Overview

The aurora is an attractive shimmering display of light from the upper atmosphere commonly seen during the night in polar regions of the earth. The northern aurora is called the Aurora Borealis, and its called Aurora Australis in the south \[13\]. Auroras occur in regions known as auroral ovals around the north and south geomagnetic poles of the earth. The auroral ovals are always in motion, expanding towards the equator or contracting towards the pole, and changing in brightness constantly. Most common colors ejected when the particles interact with the gases of the atmosphere are: red and green when it interacts with oxygen, and blue and purple when it interacts with nitrogen.

Time History of Events and Macroscale Interactions during Substorms (THEMIS) project is a NASA mission launched in 2006. The NASA THEMIS project is intended to investigate magnetospheric substorm phenomena, which are the manifestations of a basic instability of the magnetosphere and a dominant mechanism of plasma transport and explosive energy release \[5\]. The ground-based All-Sky Imager (ASI) exhibit observes the aurora from Canada to Alaska over the North American landmass
to figure out where and when the auroral substorm begins. Besides a ground-based magnetometer (GMAGs), there are 20 cameras, and together they are known as the ground-based observatories (GBOs) [14].

2.2 Classification with CNN

Computer vision is an artificial intelligence field in which computers are trained to interpret and understand the visual world. Automating the research enables searching, collecting information, and understanding and identifying patterns across large image data sets. During ongoing decades, it has exhibited that machine learning strategies are an essential and compelling resource for naturally arranging massive image data sets, for example, by letter, mind tumor, and facial recognition. However, the techniques of machine learning are not widely used within the auroral research community. The automatic classification of a large number of images captured each year will make it simpler for researchers to examine the images that are of enthusiasm for a sorted out, objective, and repeatable way.

Due to the transparent nature of the emission and thus the soft boundaries of the observed forms, machine classification of aurora is a difficult task. An extra complication is that there’s no explicit consensus on how many nightside auroral classes exist and what they are. In any case, a few automatic classifications and feature extraction procedures have been created and tried over the last two decades.

Ground-based auroral observations were instrumental in building the notion of substorms by Akasofu [15], one of the primary energy dissipation modes for the magnetosphere explained by Akasofu [16]; and Clausen [17]. A substorm, as defined
in Akasofu [15], consists of two phases: the phase of disintegration, and the phase of recovery. During the breakup, a single dim arc brightens unexpectedly, and broad night sky regions abruptly fill with bright, discreet aurora that lasts about 10 minutes or so. The aurora split darkens and becomes more patchy and scattered, entirely blurred in the long run, throughout the recovery period. It was later built up by Bargatze et al. [18], and McPherron et al., [19] that a third stage should go before the separation phase. This step was called the growth phase during this first step energy is charged into the magnetospheric tail by daytime magnetic reconnection. During this time, the arc usually is moving towards the equator, which eventually becomes the breakup arc. Since procedures in close Earth space shape the aurora, it is clear that the morphology of auroral structures begins with our understanding of magnetospheric elements from the earliest primary stage. It would be alluring, in this way, to characterize the tremendous measure of existing ground-based auroral information naturally to empower enormous factual investigations.

Initially, Syrjäsuo and Donovan [20] incorporated computer vision techniques into the interpretation of auroral images and categorized auroras into arcs, patchy auroras, omega-bands, and north-south systems, detailing their form. A lot of automatic auroral has arisen since studies, including a description of the auroral images, classification, recovery, and segmentation. Wang et al. [21] suggested that the local binary pattern operator be combined with a delicately built block partition scheme describing the auroral morphology (shape and texture). Rao et al. [22] defined the color variants of the scale-invariant feature transform features into three mutually exclusive groups for automatic all-sky camera image classification: aurora, no aurora,
and cloudy. Yang and Hu\[9\] depicted and categorized auroral images into the arc, drapery crown, radial corona, and hotspot using the Weber local descriptor. Multiple combined handcrafted features (gray, structural, and textural features) derived from auroral images, Zhong et al. \[23\] proposed a method for classifying auroral images based on the Dirichlet allocation of latent multi-feature.

Automatic auroral image detection has already used a range of methods for machine vision, pattern recognition, and computer vision, with a strong focus on hand-designed applications. Early attempts, based on sparse edges and skeletons by Syrjäsuo et al. \[24\], used a two-step classification for individual images. In this work, the pattern recognition algorithm for searching auroral arcs based on shapes and forms. Following these new approaches, k-nearest neighbor classification, and primary component shape analysis by Syrjäsuo & Donovan, \[25\] were used for auroral tracking. Here, the classified results of auroras or no auroras are compared with human expert results. This later allowed for the automatic evaluation of the auroral occurrence statistics by Syrjäsuo & Donovan \[20\], where results of auroral forms were compared with results from manual surveys. Finally, auroral highlights were further refined to fourier descriptors depending on unambiguous shape models using shape and edge identification by Syrjäsuo et al. \[26\]. Upon preparation, the automatic classification procedures used in these investigations were normally ready to recognize images indicating aurora and images accurately. However, none of the developed tools achieved wide usage, possibly due to the techniques not being general enough, had a low classification accuracy.
2.3 Current Advancements

Recently, deep learning-based algorithms have dominated most vision-based tasks due to their discriminative power. Representations of images based on CNN have attracted increasing attention from the community and demonstrate remarkable results. The main difference between CNNs and conventional machine learning methods is that CNN’s directly learn data features without the need for additional manual extraction. There are currently several CNN models for different computer vision applications, such as OverFeat [27], Alexnet [28], GoogleNet [29], ResNet [11], VGGNet [30], Inception [31], and their variants have been studied. Razavian et al [32] use pre-trained model OverFeat to extract features first and then demonstrate the off-shelf features have better performance than handcrafted features for different computer vision tasks.

Focusing on hand-crafted features from small-scale data sets and algorithms has shifted to large-scale data sets and learning machines that automatically extract feature representation from the raw data. All of these advances were rapidly integrated into the classification of auroral pictures. Wang and Yang [33] introduced AlexNet to automatically classify Yellow River Station (YRS) dayside auroral images into the arc, drapery corona, radial corona, and hotspot. However, the developed dayside tools cannot be used on nightside aurora images due to differences in auroral morphology. Clausen and Nickisch [8] used the Inception-v4 pre-trained model to automatically identify auroral images from the Oslo Auroral THEMIS (OATH) dataset into clear/no auroras, cloudy, moon, arc, diffuse and discrete. Niu et al. [34] suggested a weakly-supervised approach to semantic segmentation that would achieve mutual pixel-level localization.
of the leading local structure and classification of the auroral images at the image level. Han et al. \cite{35} have proposed multisize CNN kernels with task-specific initialization powered by eye movement to classify auroral images into the arc, radiation corona, and drapery corona.

2.4 Summary

Auroral images contain a wealth of data, but interpreting them and utilizing them effectively for classification is a huge challenge. Post-processing of auroral data sets sometimes gives an abundant amount of information regarding the sample. Innovative techniques are required to process the auroral data, hence highlighting the need for more extensive research for novel data interpretation/manipulation techniques and classification methodologies.
CHAPTER 3

METHODOLOGY

3.1 Theoretical Overview of CNN

This area provides information about the conceptual analysis of the individual parts of the proposed deep learning-based feature extraction model compared with other customary models utilized for auroral image data analysis. At the end of this chapter, we analyze the proposed architecture with the different existing standardized architecture of CNN.

3.1.1 Convolutional Neural Network

This section provides a brief description of the convolutional neural network, which includes the numbers of parts that sum up to improve the model’s efficiency and performance. The Convolutional Neural Network (CNN) is an efficient model for many computing vision fields, such as classification, image segmentation, and object detection. A supervised image classification using CNN involves data processing to find the input data features and classification of new data samples after training the machine learning model. Deep learning CNN models have different layers, such as
filters/kernels, pooling layers, and fully connected layers. Every input passes through the above CNN layers during the model training.

The primary objective of convolution is to extract and understand the characteristics of the input image. Convolutions maintain the spatial relationship between every pixel by using small squares of input data to learn image features. The sliding window is called filter or kernel in CNN terminology, and the resulting matrix is called either the feature map or the activation map. In practice, during the training process, CNN learns the values of those filters on its own [36]. Three parameters set to regulate the size of the feature map before the conversion step:

1. **Depth**: It represents the number of filters we are using for conversion. Every filter results in a tensor generated by a characteristic feature map.

2. **Stride**: It is the number of pixels by which our filter matrix is slide over the matrix of inputs. When the stride is one, we transfer one pixel of the filters at a time. At a time, the filters jump 2 pixels when the stride is 2, then have larger steps produce smaller feature maps while using a smaller step; all the information from the input functions open.

3. **Padding**: The padding of the input matrix with zeros is often convenient around the border so that the filter can be extended to the border elements of our image matrix input and preserve the image size.

When the shape of the input \((h \times w \times d)\) where \(h\) indicate the height, \(w\) indicate the width and \(d\) is the number of dimensions of input is convolved with a filter of shape \((f_h \times f_w \times d)\) where \(f_h\) is the height of the filter, \(f_w\) is the width of the filter and \(d\) is the number of dimensions of the filter, the resultant output of the
The convolution layer is of the size $(h - f_h + 1) \times (w - f_w + 1) \times 1$. The process of spatial pooling reduces the dimensionality of each map while preserving the primary data. There can be various kinds of spatial pooling: Max, Average, Sum, and so on. Pooling makes the feature dimension little and progressively reasonable, reducing the number of parameters and calculations in the system, which, in this way, control overfitting. Usually, CNN may have several layers of Convolution, pooling, normalization, and not necessarily obey the order, as shown in Figure 3.1.

![Figure 3.1: Example of Convolutional Neural Network](image)

3.1.2 Max Pooling

One of the building blocks of Convolutional Neural Network (CNN) is the pooling layers. Pooling layers in CNNs summarize the outputs of neighboring groups of neurons in the same kernel map. It also introduces invariance, reduces dimension, and prevent overfitting in CNNs.

We adopted overlapping pooling layers in our architecture. Overlapping pooling means adjacent pooling regions overlap with each other. For example, the pooling
region size is denoted as $s \times s$, and stride is denoted as $z$. If $s \geq z$, which is used in most of the CNNs, it is non-overlapping. If $s < z$, then it becomes overlapping pooling, as shown in Figure 3.2. We generally observe during training that models with overlapping pooling find it slightly more difficult to overfit [37]. We have adopted used pooling windows of size $3 \times 3$ ($s = 3$) with a stride of 1 ($z = 1$) between the adjacent windows.

![Figure 3.2: Overlapping Pooling][10]

Max pooling layers commonly used to downsample the tensor width and height, keeping the depth identical. Max pooling is a pooling operation that selects the maximum element from the region of the feature map covered by the filter. The overlapping max pool layers are similar to the max pool layers, except that the adjacent

[10]: https://example.com/figure32.png
windows over which the max is computed overlap each other. Figure 3.3 gives the pictorial description of the feature map where every $2 \times 2$ for the original matrix is pooled into a single block with max value with stride 1.

Figure 3.3: Overlapping Max Pooling

3.1.3 Reduce Overfitting

Overfitting is a common problem in the neural network, particularly when researchers do not have a large enough dataset. Overfitting happens when your model fits inside the training set too well. The model then finds it difficult to generalize to new examples that were not in the training set. There are standard methods nowadays to avoid overfitting are normalization of batches, apply regularization, and use dropout layers.
3.1.4 Batch Normalization

Batch normalization is a training technique for deep neural networks that standardizes the inputs to a layer for each mini-batch. This results in a stabilization of the learning process and a dramatic reduction in the number of training epochs required for deep networking training.

The basic idea behind batch normalization is to limit the covariate shift by normalizing the activations of each layer (transforming the inputs to be mean 0 and unit variance). This supposedly allows each layer to learn on a more stable distribution of inputs, thus accelerating the network’s training.

3.1.5 Dropout

Dropout is a method of regularization that approximates the training of a large number of neural networks in parallel with different architectures. Dropout is implemented per-layer on a neural network. If a neuron is dropped, this does not contribute to the propagation of either forward or backward. The learned weight parameters are, therefore, more robust, and are not easily overfitted. Dropout can be implemented on either or all of the network’s hidden layers, and on the visible or input layer, which cannot be used on the output layer.

The key advantage of using the dropout method is that it prevents synchronous optimization of their weights by all neurons in a layer. This adaptation, made in random groups, prevents all neurons from converging to the same objective and thus decorrelates the weights. A second property discovered for dropout application is
that the activations of the hidden units become sparse, which is also a desirable characteristic.

3.1.6 ReLU Non-Linearity

The Rectified Linear Unit (ReLU) function is essential because it isn’t saturating; if the neuron activates, the gradient is still high (equal to 1). It’s defined as \( f(x) = \max(0,x) \). The continuous updates are relatively successful as long as it is not a dead neuron. ReLU is also very quick to evaluate, follows below equation 3.1.

\[
S(x) = \begin{cases} 
  x & x \geq 0 \\
  0 & x < 0 \end{cases} 
\]  
(3.1)

Compare this with the sigmoid function, which is saturated (the gradient is minimal, whether the input is very high or very low). The gradient for inputs far from their origin is close to zero, so gradient-based learning for saturated neurons using sigmoid is slow. The vanishing gradient problem makes it difficult to know which direction the parameter should move to reduce the cost function.

3.1.7 Softmax layer and Loss function

Besides the convolutional layer and pooling layers, the network also generally implements a softmax layer as a final layer for image-related problems. It transforms the output within the probability of 0 to 1. It allows for a fundamental understanding of the output as a probability. Similarly, the softmax functions are multi-class sigmoids,
meaning that multiple class likelihoods are evaluated at once. Because a softmax function's outputs can be viewed as a probability (i.e., they will amount to 1), a softmax layer is usually the final layer used in neural network functions. It must be remembered that a layer with softmax has to have the same number of nodes as the following output. For instance, for a K-dimensional vector \( x = [x_1, x_2, ..., x_K] \), the formula for softmax is given by,

\[
f(x_i) = \frac{e^{x_i}}{\sum_{j=1}^{K} e^{x_j}}
\]

The CNNs training procedure generally consists of two steps, which are computation for feed-forward and backpropagation. In the feed-forward step, information moves from input nodes through hidden nodes to output nodes in the forward direction. A loss between outputs of CNNs and ground truth is first computed in the backpropagation step. The loss gradient function is then calculated with respect to all weights and biases in the network. Weights and biases are updated to minimize loss function through the obtained gradient. The cross-entropy loss is the widely used loss function with the following formula,

\[
H(p, q) = -\sum_{i=1}^{K} p_i \log(q_i)
\]

where, ground truth vector is \( p = [p_1, p_2, ..., p_K] \), and output vector is \( q = [q_1, q_2, ..., q_K] \) of a CNN through feed-forward computation.
3.2 CNN based Classification Methodology

This section presents a brief description of the proposed deep learning-based auroral feature extraction and data analysis framework along with the other feature extraction architectures used as a mode of comparison.

3.2.1 Proposed Method

This section presents a description of the auroral classes, the data processing, and the applied machine learning techniques. Our objective is to develop a neural network such that it can consequently classify auroral images depending on the observed features. The algorithms that we use to classify auroral images all fall into the supervised learning category, where the motive is to train a model from the labeled training data. Once the model is optimized, predictions about unseen or future data can make use of it.

The prior studies name the auroral images into diverse categories based on the auroral displays observed from the all-sky imager. For example, Syrjäsuo and Donovan [20] utilized four types to be specific no aurora, arcs, patchy aurora, and omega-bands. Later improvements by Wang and Yang [33] classified dayside auroral images of Yellow River Station that incorporate categories of names, specifically arc, drapery corona, radial corona, and hotspot. Recently, Clausen and Nickish [6] utilized a pre-trained neural network to classify auroras into arcs, diffuse, discrete, cloudy, moon, clear/no aurora from auroral THEMIS(OATH) dataset.
Labeling auroral pictures may be a challenging task since no two auroral events are alike. The aurora could be an exceptionally dynamic phenomenon that changes quickly and shows differently depending on geomagnetic conditions. The proposed set of auroral names was simple to distinguish beneath distinctive geomagnetic conditions and pertinent to most of the auroral images, despite the blend of classes existed.

We chose to add five labels based on the sky conditions into the clear sky with auroras, cloudy with the moon, cloudy, clear sky with the moon, and clear sky.

<table>
<thead>
<tr>
<th>Label</th>
<th>Class</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>Clear sky with aurora</td>
<td>These images look like a clear, obvious aurora of diverse varieties</td>
</tr>
<tr>
<td>1</td>
<td>Cloudy with the moon</td>
<td>The sky in these images are cloudy with the presence of moonlight</td>
</tr>
<tr>
<td>2</td>
<td>Cloudy</td>
<td>The sky in these images is absolutely covered with clouds.</td>
</tr>
<tr>
<td>3</td>
<td>Clear sky with the moon</td>
<td>The image shows a clear sky (stars are visible) with presence of moonlight</td>
</tr>
<tr>
<td>4</td>
<td>Clear sky</td>
<td>The images show a clear sky, (stars are visible) with no aurora present.</td>
</tr>
</tbody>
</table>

The reason for manual labeling of auroral images is to avoid ambiguity during training. The sample images of each category used to train the model are listed in Figure 3.4.

The auroral images undergo a pre-processing step before we feed 128 x 128 resized images as input to the neural network. The neural network training is an
Figure 3.4: Sample Images of each Categories. (a) Clear sky with aurora (b) Cloudy with the moon (c) Cloudy (d) Clearsky with the moon (e) Clearsky

iterative process that advances neurons during the forward pass and updates parameters during the backward pass until losses are minimized at each iteration in a forward pass. The network comprises of five convolutional layers, four pooling layer, and four fully connected layers. A convolutional layer extricates the highlights utilizing learnable filters over feature maps from the past layer. The output feature map of each filter stacked on top of each other, whose depth is the number of filters utilized within the layer. Also, in each convolutional layer, the values of neurons are transformed with a nonlinear activation function ReLU.

A pooling layer makes use of the max-pooling feature for downsampling activation maps. In the pooling area, the max-pooling function selects the maximum neuron value (defined by the size of a pooling filter) and disregards others. We make use of overlapping pooling operation in the pooling layer. The dropout method is used in the first and second fully connected layers. The final layer produces probabilities corresponding to five classes. We use cross-entropy loss, Adam Optimizer, that was
developed using deep learning framework Pytorch. Table 3.2 defines the architecture used in our research.

### Table 3.2: Architecture of our Network

<table>
<thead>
<tr>
<th>Hyperparameter</th>
<th>Conv1</th>
<th>Pool1</th>
<th>Conv2</th>
<th>Pool2</th>
<th>Conv3</th>
</tr>
</thead>
<tbody>
<tr>
<td>Number of filters</td>
<td>64</td>
<td>0</td>
<td>150</td>
<td>0</td>
<td>256</td>
</tr>
<tr>
<td>Filter size</td>
<td>8 x 8</td>
<td>3 x 3</td>
<td>5 x 5</td>
<td>3 x3</td>
<td>3 x3</td>
</tr>
<tr>
<td>Stride</td>
<td>2</td>
<td>2</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>Padding</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Hyperparameter</th>
<th>Pool3</th>
<th>Conv4</th>
<th>Conv5</th>
<th>Pool4</th>
</tr>
</thead>
<tbody>
<tr>
<td>Number of filters</td>
<td>0</td>
<td>256</td>
<td>128</td>
<td>0</td>
</tr>
<tr>
<td>Filter size</td>
<td>2 x 2</td>
<td>2 x 2</td>
<td>2 x 2</td>
<td>3 x3</td>
</tr>
<tr>
<td>Stride</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>Padding</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
</tbody>
</table>

#### 3.2.2 Methodology for Comparison

This section presents a brief description of the proposed feature extraction and data analysis framework along with the other feature extraction architectures used as a mode of comparison.

#### 3.2.2.1 Proposed Architecture

Once we obtain the images from the THEMIS database, we pre-process auroral images and resize the images before training. Then, resized images then become the input to the network. Such developed areas are network entry. Training CNN is a method in which neurons advance through the forward pass, updates the network parameters (for example, weights of convolutional filters) during the backpropagation pass, and minimizes the losses during the forward iteration measured.
The convolution layer extracts features across feature maps from the feature maps created in the previous layer. The feature extraction of the convolution layer is possible only because of the learnable filters. The output of the filter is stacked on top of each other to create a three-dimensional output. Depending on the total number of filters, the depth can be calculated.

The number of filters in the preceding layer \((1−1)\) corresponds to the depth of the map stack input feature; \(b^k\) is the \(k^{th}\) filter bias. We Ignore the outermost sum over the depth when \(l = 0\) (the first convolutional layer), and transform the values of neurons into each convolutional layer with a nonlinear activation function. Eventually, we chose ReLU as the role of activation within the convolution layer. Downsampling of the feature maps happens in a pooling layer by using the max-pooling method. Here, the max-pooling method selects the maximum value of the neuron in the pooling region and ignores the remaining neuron values.

We get a single vector when the first fully connected layer of CNN flattens from the last layer of the normalization layer. A dropout method is then applied to the first and second fully connected layers prevents all the neurons from optimizing the weights. The last fully-connected layer predicts the probability of the corresponding that it belongs to the five classes. The proposed architecture has five convolutional layers, four pooling layer, and six fully connected layers. The pictorial form of the proposed architecture is as follows in Figure 3.5.
3.2.2.2 Alexnet

AlexNet architecture is composed of five convolutional layers and three fully Connected Layers. In an image, multiple fully convolutional kernels (a.k.a filters) extract the features. There are typically several kernels of the same size in a single convolutional sheet. AlexNet’s first Conv Layer, for example, includes 96 filters of 11x11x3 kernel size. Remember that the kernel width and height are generally equal, and the depth is the same as the number of channels.

The first two layers of the convolution are followed by the layers of Overlapping Max Pooling, which we discuss next. There is a direct connection between the third,
fourth, and fifth convolutional layers. Following the fifth convolutional layer is an overlapping max-pooling layer, whose output goes into a series of two fully connected layers. After all the convolution and completely connected layers, the second fully connected layer feeds into a softmax classifier with 1000 class labels \(^2\). ReLU applies nonlinearity. Before doing pooling, the ReLU nonlinearity of the first and second convolution layers is followed by a step of local normalization. The below Figure 3.6 provides a generic form of alexnet used to classify 1000 class labels. In our work, we have made sure that the standard alexnet model is used to classify the input data into five classes based on sky conditions.

**Figure 3.6:** Alexnet Architecture \(^2\)
3.2.2.3 VGG

Although previous AlexNet derivatives concentrated in the first convolutionary layer on smaller window sizes and strides, Visual Geometry Group (VGG) discusses another essential feature of CNNs: width. Let us go over VGG’s architecture:

1. **Input**: VGG takes in an RGB image of 224x224 pixels. To keep the input image size consistent, the authors cropped out the 224x224 patch center in each image for the ImageNet competition.

2. **Convolutional Layers**: For VGG, the convolutionary layers use a small receptive field (3x3, the smallest amount always capturing left/right and up/down). Also, 1x1 convolution filters function as a linear input transformation followed by a ReLU unit. The convolution stage is set at 1 pixel so that after convolution, the spatial resolution is retained.

3. **Fully-Connected Layers**: VGG has three fully interconnected layers: the first two have 4096 channels each, and the third has 1000 channels, 1 for each.

4. **Hidden Layers**: All hidden layers of VGG are using ReLU (a breakthrough from AlexNet that is reducing training time). Local Response Normalization (LRN) is usually not used by VGG since LRN increases memory usage and training time without any specific improvement inaccuracy.\(^3\) The pre-trained VGG-16 and VGG-19 architecture are defined, as shown in Figure 3.7.

In our work, we have made sure that both of the pre-trained VGG-16 and VGG-19 model is taken into consideration. The pre-trained vgg models classify 1000
classes, but in our case model classifies the images from the training dataset into five classes based on the sky conditions.

Figure 3.7: VGG Architecture

3.2.2.4 Resnet

A residual neural network (ResNet) is a form of artificial neural network (ANN) that builds on constructs recognized from pyramidal cells within the cerebral cortex. This is achieved through residual neural networks using skip links or shortcuts to leap over a few layers. Typical ResNet models are implemented with double or triple layer skips containing nonlinearities (ReLU) and intermediate batch normalization. The skip weights can be learned with a new weight matrix; such models are called HighwayNets. Multiple parallel skips in the models are known as DenseNets. A non-residual network can be represented in the sense of the residual neural networks as a plain network [38].

One reason for skipping over layers is to prevent vanishing gradients, by reusing activations from a previous layer before the neighboring layer knows as weights. In
the most straightforward case, just the weights for the nearby layer’s association are adjusted, with no specific weights for the upstream layer. This works best when stepping over a single nonlinear layer, or when all of the intermediate layers are linear. If not, then for the missed link (a HighwayNet should be used), a specific weight matrix should be taught.

Skipping connections simplifies the network, using fewer layers at the initial stages of training. This speeds learning by the effect of vanishing gradients, as there are fewer layers to spread through. The network then slowly restores the layers that have been missed as they learn the feature space. As all layers extended toward the end of the training, it stays closer to the manifold and absorbs more quickly. A neural network with no remaining parts explores more space on the function. That makes it more vulnerable to disturbances that cause it to leave the manifold and requires extra retrieval of training data.

The pre-trained ResNet architecture is defined as in Figure 3.8. In our work, we have made sure that both of the pre-trained ResNet-32 and ResNet-20 model is taken into consideration. Here, the pre-trained ResNet models are used to classify the images from the training data set into five classes based on sky conditions.
3.3 Experimental Design

We perform all the experiments using Pytorch, open-source platform developed by Facebook’s AI research lab. Pytorch is an open-source and free software released under the Modified BSD license. It is accelerated with Nvidia GeForce GTX 1080.
CHAPTER 4

EXPERIMENTAL EVALUATION

This section describes the technique involved in data pre-processing and feature extraction using machine learning. Further, it analyzes classification scores based on the aurora data set for various deep-learning models. The results of the proposed architecture are compared and validated with the results of traditional architectural models used for auroral classification, and the efficiency of the model is evaluated.

4.1 Data Preprocessing

In this section, we discuss how data is pre-processed before the information is fed into the model. The instrumentation and data sub-section presents a brief overview of how the THEMIS GBO ASI data is stored and maintained for public outreach and educational purposes.

4.1.1 Instrumentation and Data

Time History of Events and Macroscale Interactions During Substorms (THEMIS) is a mission launched by NASA in 2006. THEMIS requires twenty white light All-Sky Imager (ASIs) to be deployed within a continent-wide array.
An array of stations composed of 20 all-sky imagers and 30 plus magnetometers was developed and deployed for the broad coverage of the night-side magnetosphere in the Northern American continent from Alaska to Labrador, as shown in the Figure 4.1.

Figure 4.1: Map of North America with GBO Station Names, Fields of View of the All Sky Imagers with Locations. [5]

The ASI will allow researchers to observe the aurora over a large segment of the auroral oval, with goals of one kilometer. The full-resolution images of 256 x 256 pixels are transferred via hard-disk and become available about 3 - 5 months after data collection.

The ASIs capture northern lights images and movies by looking over each ASI position from horizon to horizon up into the sky. It’s done using a ’fish-eye’ lens [39]. The cameras capture black and white images and capture all the visible light from
the aurora, allowing even very weak auroras to be recorded, which our eyes cannot see at 1-second exposure.

The stations return compact images supposed to be two ”thumbnails” to focal databases, one at UC Berkeley and the other at U Calgary, Canada. The complete images are recorded on hard drive storage disks that are returned to the two databases to replicate the information at the station. The ASIs are white light (unfiltered) cameras, using a very efficient (F/0.95) fisheye optical system with a highly sensitive charging-coupled camera device (CCD). All information is made available through web programs or as downloadable Common Data Format (CDF) information records for open use by researchers [5].

4.1.2 Data Pre-Processing

In this section, we briefly discuss the data pre-processing of the CDF files before the full resolution all-sky imager is fed to the convolutional neural network.

4.1.3 SpacePy

SpacePy is a space science-oriented python package that facilitates fundamental data analysis, modeling, and visualization. It builds on the capabilities of well-known packages, including NumPy and Matplotlib [40]. The goals of SpacePy emphasize on:

- Obtain the data quickly
- Perform complicated analysis easier
- Harness the power of python
pyCDF is python access to NASA CDF Library. It provides a "pythonic" interface to the NASA CDF library. A NASA CDF only has a single 'layer' by definition. In other words, a CDF includes a series of records (stored variables of different types) and a set of attributes that are either global or local in scope [40].

In our experiment, we used data originated from UC Berkeley THEMIS all-sky imager network between 2006 and 2019. The images are resized to 128 x 128 pixels. We have labeled the night time auroral images manually into five classes: clear sky with auroras, clouds with the moon, cloudy, clear sky with moon, and clear sky based on sky conditions. The proposed set of auroral images classifies auroral images according to conditions of being clear, cloudy, and auroras. A clear data set was created by excluding ambiguous auroral forms, unwanted features, and disagreeable labels, and by taking into account only agreeable labels. The data is divided into data set for training and testing. In total, we have 3100 images in the final data set, which includes 2500 training and 600 as independent test data to gauge classifier performance. The number of images acquired each year is given in Table 4.1. The Tables 4.2, 4.3, 4.4, and 4.5 gives the information on how many images are collected using different stations from 2006 to 2019 for the data set creation to conduct our experiment.
Table 4.1: Number of Images Per Year Based on the Sky Conditions

<table>
<thead>
<tr>
<th>Year</th>
<th>Number of images</th>
</tr>
</thead>
<tbody>
<tr>
<td>2006</td>
<td>274</td>
</tr>
<tr>
<td>2007</td>
<td>342</td>
</tr>
<tr>
<td>2008</td>
<td>500</td>
</tr>
<tr>
<td>2009</td>
<td>321</td>
</tr>
<tr>
<td>2010</td>
<td>285</td>
</tr>
<tr>
<td>2011</td>
<td>288</td>
</tr>
<tr>
<td>2012</td>
<td>146</td>
</tr>
<tr>
<td>2013</td>
<td>74</td>
</tr>
<tr>
<td>2014</td>
<td>169</td>
</tr>
<tr>
<td>2015</td>
<td>45</td>
</tr>
<tr>
<td>2016</td>
<td>74</td>
</tr>
<tr>
<td>2017</td>
<td>84</td>
</tr>
<tr>
<td>2018</td>
<td>76</td>
</tr>
<tr>
<td>2019</td>
<td>422</td>
</tr>
<tr>
<td>Total</td>
<td>3100</td>
</tr>
</tbody>
</table>

Table 4.2: Count of the Images Used in our Experiment from the ASI Stations KIAN, MCGR, FYKN, GAKO, INUV.

<table>
<thead>
<tr>
<th>Year</th>
<th>KIAN</th>
<th>MCGR</th>
<th>FYKN</th>
<th>GAKO</th>
<th>INUV</th>
</tr>
</thead>
<tbody>
<tr>
<td>2006</td>
<td>0</td>
<td>2</td>
<td>19</td>
<td>0</td>
<td>57</td>
</tr>
<tr>
<td>2007</td>
<td>0</td>
<td>9</td>
<td>3</td>
<td>76</td>
<td>32</td>
</tr>
<tr>
<td>2008</td>
<td>3</td>
<td>0</td>
<td>20</td>
<td>87</td>
<td>25</td>
</tr>
<tr>
<td>2009</td>
<td>3</td>
<td>0</td>
<td>14</td>
<td>45</td>
<td>39</td>
</tr>
<tr>
<td>2010</td>
<td>4</td>
<td>7</td>
<td>4</td>
<td>30</td>
<td>33</td>
</tr>
<tr>
<td>2011</td>
<td>11</td>
<td>6</td>
<td>22</td>
<td>37</td>
<td>22</td>
</tr>
<tr>
<td>2012</td>
<td>1</td>
<td>0</td>
<td>17</td>
<td>26</td>
<td>3</td>
</tr>
<tr>
<td>2013</td>
<td>2</td>
<td>2</td>
<td>1</td>
<td>7</td>
<td>1</td>
</tr>
<tr>
<td>2014</td>
<td>18</td>
<td>5</td>
<td>5</td>
<td>0</td>
<td>5</td>
</tr>
<tr>
<td>2015</td>
<td>3</td>
<td>0</td>
<td>2</td>
<td>2</td>
<td>2</td>
</tr>
<tr>
<td>2016</td>
<td>4</td>
<td>0</td>
<td>5</td>
<td>4</td>
<td>12</td>
</tr>
<tr>
<td>2017</td>
<td>2</td>
<td>10</td>
<td>3</td>
<td>1</td>
<td>3</td>
</tr>
<tr>
<td>2018</td>
<td>5</td>
<td>3</td>
<td>2</td>
<td>0</td>
<td>7</td>
</tr>
<tr>
<td>2019</td>
<td>11</td>
<td>19</td>
<td>0</td>
<td>6</td>
<td>49</td>
</tr>
</tbody>
</table>
Table 4.3: Count of the Images Used in our Experiment from the ASI Stations WHIT, FSMI, TALO, SNAP, FSIM.

<table>
<thead>
<tr>
<th>Year</th>
<th>WHIT</th>
<th>FSMI</th>
<th>TALO</th>
<th>SNAP</th>
<th>FSIM</th>
</tr>
</thead>
<tbody>
<tr>
<td>2006</td>
<td>35</td>
<td>31</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>2007</td>
<td>28</td>
<td>2</td>
<td>0</td>
<td>0</td>
<td>13</td>
</tr>
<tr>
<td>2008</td>
<td>61</td>
<td>25</td>
<td>0</td>
<td>0</td>
<td>36</td>
</tr>
<tr>
<td>2009</td>
<td>15</td>
<td>5</td>
<td>9</td>
<td>0</td>
<td>8</td>
</tr>
<tr>
<td>2010</td>
<td>16</td>
<td>6</td>
<td>19</td>
<td>12</td>
<td>4</td>
</tr>
<tr>
<td>2011</td>
<td>19</td>
<td>3</td>
<td>10</td>
<td>15</td>
<td>10</td>
</tr>
<tr>
<td>2012</td>
<td>1</td>
<td>8</td>
<td>0</td>
<td>1</td>
<td></td>
</tr>
<tr>
<td>2013</td>
<td>5</td>
<td>12</td>
<td>1</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>2014</td>
<td>5</td>
<td>19</td>
<td>12</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>2015</td>
<td>3</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>2016</td>
<td>5</td>
<td>7</td>
<td>0</td>
<td>0</td>
<td>4</td>
</tr>
<tr>
<td>2017</td>
<td>1</td>
<td>1</td>
<td>6</td>
<td>0</td>
<td>12</td>
</tr>
<tr>
<td>2018</td>
<td>4</td>
<td>1</td>
<td>2</td>
<td>2</td>
<td>2</td>
</tr>
<tr>
<td>2019</td>
<td>41</td>
<td>35</td>
<td>35</td>
<td>0</td>
<td>12</td>
</tr>
</tbody>
</table>

Table 4.4: Count of the Images Used in our Experiment from the ASI Stations ATHA, TPAS, RANK, GILL, PINA, GBAY

<table>
<thead>
<tr>
<th>Year</th>
<th>ATHA</th>
<th>TPAS</th>
<th>RANK</th>
<th>GILL</th>
<th>PINA</th>
<th>GBAY</th>
</tr>
</thead>
<tbody>
<tr>
<td>2006</td>
<td>32</td>
<td>62</td>
<td>31</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>2007</td>
<td>10</td>
<td>66</td>
<td>8</td>
<td>27</td>
<td>37</td>
<td>0</td>
</tr>
<tr>
<td>2008</td>
<td>3</td>
<td>89</td>
<td>51</td>
<td>31</td>
<td>5</td>
<td>6</td>
</tr>
<tr>
<td>2009</td>
<td>17</td>
<td>33</td>
<td>9</td>
<td>17</td>
<td>17</td>
<td>11</td>
</tr>
<tr>
<td>2010</td>
<td>0</td>
<td>16</td>
<td>31</td>
<td>7</td>
<td>16</td>
<td>20</td>
</tr>
<tr>
<td>2011</td>
<td>14</td>
<td>16</td>
<td>29</td>
<td>7</td>
<td>8</td>
<td>2</td>
</tr>
<tr>
<td>2012</td>
<td>18</td>
<td>13</td>
<td>17</td>
<td>12</td>
<td>8</td>
<td>0</td>
</tr>
<tr>
<td>2013</td>
<td>5</td>
<td>0</td>
<td>5</td>
<td>5</td>
<td>11</td>
<td>2</td>
</tr>
<tr>
<td>2014</td>
<td>8</td>
<td>12</td>
<td>10</td>
<td>11</td>
<td>12</td>
<td>11</td>
</tr>
<tr>
<td>2015</td>
<td>6</td>
<td>4</td>
<td>5</td>
<td>3</td>
<td>2</td>
<td>0</td>
</tr>
<tr>
<td>2016</td>
<td>0</td>
<td>0</td>
<td>8</td>
<td>5</td>
<td>0</td>
<td>5</td>
</tr>
<tr>
<td>2017</td>
<td>1</td>
<td>0</td>
<td>2</td>
<td>5</td>
<td>7</td>
<td>5</td>
</tr>
<tr>
<td>2018</td>
<td>4</td>
<td>3</td>
<td>7</td>
<td>4</td>
<td>3</td>
<td>2</td>
</tr>
<tr>
<td>2019</td>
<td>10</td>
<td>5</td>
<td>48</td>
<td>6</td>
<td>14</td>
<td>23</td>
</tr>
</tbody>
</table>
Table 4.5: Count of the Images Used in our Experiment from the ASI Stations KAPU, PGEQ, SNKQ, CHBG, KUUJ, NRSQ

<table>
<thead>
<tr>
<th>Year</th>
<th>KAPU</th>
<th>PGEQ</th>
<th>SNKQ</th>
<th>CHBG</th>
<th>KUUJ</th>
<th>NRSQ</th>
</tr>
</thead>
<tbody>
<tr>
<td>2006</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>2007</td>
<td>3</td>
<td>5</td>
<td>10</td>
<td>13</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>2008</td>
<td>4</td>
<td>1</td>
<td>22</td>
<td>19</td>
<td>4</td>
<td>0</td>
</tr>
<tr>
<td>2009</td>
<td>13</td>
<td>0</td>
<td>5</td>
<td>25</td>
<td>6</td>
<td>7</td>
</tr>
<tr>
<td>2010</td>
<td>2</td>
<td>9</td>
<td>9</td>
<td>3</td>
<td>15</td>
<td>19</td>
</tr>
<tr>
<td>2011</td>
<td>10</td>
<td>0</td>
<td>14</td>
<td>0</td>
<td>10</td>
<td>18</td>
</tr>
<tr>
<td>2012</td>
<td>9</td>
<td>0</td>
<td>9</td>
<td>0</td>
<td>3</td>
<td>0</td>
</tr>
<tr>
<td>2013</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>6</td>
<td>1</td>
<td>5</td>
</tr>
<tr>
<td>2014</td>
<td>15</td>
<td>0</td>
<td>6</td>
<td>15</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>2015</td>
<td>5</td>
<td>0</td>
<td>4</td>
<td>2</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>2016</td>
<td>5</td>
<td>0</td>
<td>10</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>2017</td>
<td>8</td>
<td>0</td>
<td>12</td>
<td>2</td>
<td>3</td>
<td>0</td>
</tr>
<tr>
<td>2018</td>
<td>22</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>2</td>
</tr>
<tr>
<td>2019</td>
<td>24</td>
<td>0</td>
<td>43</td>
<td>0</td>
<td>41</td>
<td>0</td>
</tr>
</tbody>
</table>

4.2 Training and Testing

As the network progresses through epochs, the network learned from the feature map and is updating hyperparameters with slowly increasing accuracy and achieving minimal losses, as shown in Table 4.6.

Compared to the processing convolution layer, the number of filters at the first layers of the convolution layer is less, which has reduced filter size. The main reason for having stride in the network is to provide shifts in convolution steps, which affect the size of the map of the output feature. Following the stride is the padding value, which adds zero-valued neurons around the map of the input features.

During testing, the network is fed with the collection of unfed data sets. The purpose of testing is to evaluate the performance of the network, which has learned
the features during training. The test data set is used to generate the results of our assessments. The loss function used in the model is Cross Entropy.

**Table 4.6: Hyperparameters of the Proposed Architecture Used for Classification**

<table>
<thead>
<tr>
<th>Hyperparameter</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Learning rate</td>
<td>1e-4</td>
</tr>
<tr>
<td>Weight decay</td>
<td>0.00005</td>
</tr>
<tr>
<td>Optimizer</td>
<td>Adam Optimizer</td>
</tr>
</tbody>
</table>

### 4.3 Evaluation Results

In this section, we use the methods described earlier to select a model that best performs the classification function. The network functioned well and avoided overfitting because the loss value did not fluctuate. Loss value narrowed to a minimum until it converged, and did not diverge.

The performance on the aurora data set is evaluated by the classification scores for different models related to deep learning. We used the confusion matrix to statistically test the accuracy of the model we learned from the classification. The correctly classified test images identified in True Positive (TP) and True Negative (TN) and misclassified test data were defined as false positives (FP) and false negatives (FN) during testing of model’s results. The study assesses the classification scores by measuring Precision, Recall, and F1 Score.

**Precision:**

Precision is defined as the ratio of correctly predicted positive observations of the total predicted positive observations. It measures the classifier’s ability to not label
positive samples as negative.

\[ Precision = \frac{\text{TruePositive}}{\text{TruePositive} + \text{FalsePositive}} \]

Recall:

A recall is the ratio of correctly predicted positive observations to all observations in actual class - yes. It evaluates the classifier’s ability to look for positive samples.

\[ Recall = \frac{\text{TruePositive}}{\text{TruePositive} + \text{FalseNegative}} \]

F1 Score:

F1 Score acts as a weighted average of precision and recall. Therefore, this Score takes both false positives and false negatives into account.

\[ F1 = 2 \times \frac{\text{Precision} \times \text{Recall}}{\text{Precision} + \text{Recall}} \]

The five deep learning-based models considered: VGG-16, VGG-19, Alexnet, Resnet-32, Resnet-20, and the proposed architecture to evaluate deep learning models. The results are shown in Table 4.7 below.
Table 4.7: Precision, Recall and F1 score for Different Classifiers on Auroral Dataset

<table>
<thead>
<tr>
<th>Architecture</th>
<th>Precision(%)</th>
<th>Recall(%)</th>
<th>F1 Score(%)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Proposed Architecture</td>
<td>96.38 ± 0.30</td>
<td>96.30 ± 0.30</td>
<td>96.35 ± 0.20</td>
</tr>
<tr>
<td>Alexnet</td>
<td>95.043 ± 0.2</td>
<td>94.833 ± 0.1</td>
<td>94.832 ± 0.02</td>
</tr>
<tr>
<td>Resnet32</td>
<td>95.615 ± 0.25</td>
<td>95.399 ± 0.1</td>
<td>95.398 ± 0.4</td>
</tr>
<tr>
<td>Resnet20</td>
<td>95.71 ± 0.04</td>
<td>95.83 ± 0.2</td>
<td>95.8 ± 0.1</td>
</tr>
<tr>
<td>VGG16</td>
<td>96.01 ± 0.06</td>
<td>95.1 ± 0.02</td>
<td>95.55 ± 0.02</td>
</tr>
<tr>
<td>VGG19</td>
<td>94.53 ± 0.2</td>
<td>94.16 ± 0.03</td>
<td>94.11 ± 0.02</td>
</tr>
</tbody>
</table>

It is essential to know the number of correctly classified data per each class to understand the performance of the model. The average classification accuracy of the proposed architecture is 96.33 ± 0.44%.

4.3.1 Confusion Matrix

A confusion matrix is a summary of prediction outcomes over a classification task. The number of correct and wrong predictions is summed up and broken down by class. That’s the key to the confusion matrix. The matrix of confusion shows how your model of classification gets confused when it makes predictions. It gives us insight into a classifier’s errors and, more importantly, the types of mistakes made.

We computed the confusion matrix for our network, as shown in Figure 4.2. From Figure 4.2, it is clear that all the auroral classes are well-known and identified correctly by the network.
4.4 Feature Extraction of Auroral Images

The experiment’s underlying aim is to understand what features the network learns during the training phase. The feature map of each of the convolution layer has been visualized. The feature maps are image grids in which the total number of filters in each convolutional layer corresponds to the number of images. The below Figure 4.3 shows the randomly fed input to the network.
The critical point to be noted is that the feature map tends to become compact and localized as the network converges. The non-linear activation function ReLU is considered in our architecture. The downside of using ReLU is "dead neurons," where each of the dead neurons corresponds to the feature map that failed to extract the features from the input. The Figure 4.4, 4.5, 4.6, 4.7, and 4.8 visualizes the feature map of the convolutional layers, which provides a brief picture of the filters that failed to extract the input features. Since the learning rate decayed properly, there was no problem because of dead neurons.
Figure 4.4: Feature map of Convolutional Layer-1

Figure 4.5: Feature map of Convolutional Layer-2
Figure 4.6: Feature map of Convolutional Layer-3

Figure 4.7: Feature map of Convolutional Layer-4
4.5 Evaluation of activation functions

In this section, we compared the sigmoid and ReLU activation functions for the activation. As shown in the figure, sigmoid slows down the network’s learning process. In contrast, ReLU yields the highest accuracy and the lowest loss value. In CNN, it preserves the receptive neuron field. The ReLU loss is diminished exponentially as the network converges over epochs. The Figure 4.9 compares the difference with the loss, and Figure 4.10 of two activation functions ReLU and sigmoid.
Figure 4.9: Loss Comparison of ReLU vs Sigmoid

Figure 4.10: Accuracy Comparison of ReLU vs Sigmoid
4.6 Result Comparison

The accuracy curve of five deep learning models are compared. The proposed architecture stands out when compared to other architectures used in comparison as shown in the Figure 4.11.

Figure 4.11: Accuracy Comparison plot
4.7 Summary

In this chapter, we evaluated the results of the proposed architecture. The experiment results of proposed architecture is compared with traditional models such as the VGG-16, VGG-19, Alexnet, Resnet-20, and Resnet-32 architectures. Experiments were carried out to classify the images into five categories: clear sky with auroras, cloudy with the moon, cloudy, clear sky with the moon, and clear sky based on sky conditions. The results of the proposed architecture outperform conventional models. Ideally, the network will classify the most dominant feature. There must be sufficient sample numbers, as small numbers of certain classes cause non-convergence. The network visualization helps to understand the ease of knowing how the network learns the features as they converge. Overall, we found that our model’s average classification accuracy is 96.33 ± 0.04 %.
CHAPTER 5

CONCLUSION AND FUTURE ENHANCEMENTS

The thesis work gives a brief idea of a deep neural network developed to classify the auroral images. We have manually labeled the full resolution all-sky imager into five categories based on the sky conditions. We have also visualized the convolutional layers to see how the neural network learns the features. We have further discussed the performance and effectiveness of training and classification.

This study can easily classify THEMIS all-sky imagers that belong to five categories: clear sky with auroras, cloudy with the moon, cloudy, clear sky with the moon and clear sky based on the sky conditions. It is also observed that the proposed architecture has outperformed other classical architectures VGG-16, VGG-19, Alexnet, Resnet-32, Resnet-20. The hyperparameter values are aligned in this work based on intermediate training results and the analysis of characteristics. We have even visualized the feature map of the convolutional neural network to know how the network learns the extracted features as the network convolves deeper.

The research has opened the window for further improvements to the proposed architecture. To further improve the model’s performance, we can fine-tune the hyperparameters. The future endeavors in aurora classification should investigate the
dimension space using different cameras and auroral events. Further improvements can be made to the classifier by including the timing dimension using Recurrent Neural Network (RNN). As we manually label the auroral images, it is a tedious and time-consuming process. So, semi-supervised or unsupervised learning techniques can be incorporated to save time involved in the data set preparation. The inclusion of time dimension allows classifying the labels with temporal behavior. We can further improve the labeling categories for fine-tuning the categories (arcs, patches, etc.) so that the higher precision for the classifier can be achieved.
chapter 6

references


[40] Spacepy documentation, 2019.