# Deep Learning Algorithm for Satellite Imaging Based Cyclone Detection

Snehlata Shakya, Sanjeev Kumar, and Mayank Goswami

***Abstract*—Satellite images are primary data in weather prediction modeling. Deep learning-based approach, a viable candidate for automatic image processing, requires large sets of annotated data with diverse characteristics for training purposes. Accuracy of weather prediction improves with data having a relatively dense temporal resolution. We have employed interpolation and data augmentation techniques for enhancement of the temporal resolution and diversifications of characters in a given dataset. Algorithm requires classical approaches during preprocessing steps. Three optical flow methods using 14 different constraint optimization techniques and five error estimates are tested here. The artificially enriched data (optimal combination from the previous exercise) are used as a training set for a convolutional neural network to classify images in terms of storm or nonstorm. Several cyclone data (eight cyclone datasets of a different class) were used for training. A deep learning model is trained and tested with artificially densified and classified storm data for cyclone classification and locating the cyclone vortex giving minimum 90% and 84% accuracy, respectively. In the final step, we show that the linear regression method can be used for predicting the path.**

***Index Terms*—Miscellaneous applications, optical data.**

## I. Introduction

### A. Remote Sensing (RS)

RS APPLICATIONS, mainly, via satellite imagery has expanded from conventional meteorology, geological exploration, oceanography toward homeland security, urban planning, ecology and several other novel unconventional fields. Cost-effective unmanned aerial vehicle (UAV) and weather balloon approaches have shared the burden but suffer from limitations such as 1) low elevation point for imaging, 2) stability issues under bad weather conditions, and 3) dependence on navigation satellites. Meteorological applications, especially weather forecasting for disaster readiness, yet, requires dedicated but costly satellite infrastructure. Routine scheduling of multispectral satellite imaging requires optimized schemes fulfilling different reward opportunities such as operating time windows, changeover efforts between two consecutive imaging tasks, cloud-coverage effects, etc. [1], [2]. Sometimes expected data often are not available at the landscape scale in abundance [3]. Trading-off between several such engineering factors limits the temporal resolution of imaging dataset, typically an hour. Resource optimization equally affects temporal resolution when imaging is performed using UAV or weather balloon.

### B. Image Processing Framework

The sparsely acquired imaging datasets comprise of limited temporal resolution that may affect the accuracy of the analysis [4]. Expert individuals can perceive and estimate missing information (such as if a set of images depict storm, a pathway of clouds, location of vortex, etc.) just seeing consecutive image frames. However, the accuracy of analysis again depends on this person's experience and off course temporal resolution of timeframes. Although classical image processing techniques are proven useful but the element of humanlike perception can only be replicated via an artificial neural network (ANN) based image processing algorithm. Several fields of research and applications have exploited this direction but state of the art for weather prediction analysis still is under development [4]. Few examples such as rainfall predictions using limited data setting [3], estimation of hydrological variables to forecast the runoff at ungauged river basins [5], air quality index estimation and prediction [6], analyzing and predicting an individual's movements/locations [7], optical flow based interpolation for structure-preservation using tomography images for improving data quality [8], [9], precipitation now casting as a spatio–temporal sequence forecasting problem [10], etc., are based on ANN.

Studies have reported weather forecasting problems using machine-learning algorithms [10], [11]. Deep learning (DL) algorithms, which learn the characteristic features in a hierarchical manner, have been introduced into the RS community due to the availability of data. DL has a wide variety of applications in RS including image preprocessing, pixel-based classifications, target identification, and scenario understanding. In one of the recent surveys, the superiority of DL algorithm (due to feature learning abilities) is shown to outperform existing commonly used image processing algorithm in hybrid field of agriculture and RS [12]. It is also shown that an outcome can be enhanced with spatio–temporal interpolation by hybridizing discriminatively trained predictive models with a deep neural network [11].

Manuscript received November 3, 2019; revised December 31, 2019; accepted January 18, 2020. Date of publication January 31, 2020; date of current version March 2, 2020. *(Corresponding author: Snehlata Shakya.)*

Snehlata Shakya is with the Department of Mathematics, Indian Institute of Technology Roorkee, Roorkee 247667, India, and also with the Department of Clinical Physiology, Lund University, 221 00 Lund, Sweden (e-mail: sneh022@gmail.com).

Sanjeev Kumar is with the Department of Mathematics, Indian Institute of Technology Roorkee, Roorkee 247667, India (e-mail: malikfma@iitr.ac.in).

Mayank Goswami is with the Divyadrishti Imaging Laboratory, Department of Physics, Indian Institute of Technology Roorkee, Roorkee 247667, India (e-mail: mayankfph@iitr.ac.in).

This article has supplementary downloadable material available at http://ieeexplore.ieee.org, provided by the authors.

Digital Object Identifier 10.1109/JSTARS.2020.2970253

A deep neural network with stacked denoising autoencoders is found comparatively better for predicting air temperature when compared with standard ANNs [13].

For image preprocessing purposes, a specific deep network such as a deconvolution network [14] and a sparse de-noising auto-encoder [15] are constructed. Pixel-based classification is already employed in the field of geoscience and RS. Aspects of handcrafted feature description [15], discriminative feature learning [16], and powerful classifier designing [17] are successfully tested. DL methods, thus, are made well suited for the extraction of low-level features with a high frequency, such as edges, contours, and outlines of objects, shape, size, color, and rotation angle of the targets [18]. Training of such an algorithm requires a significant amount (quantitative reference) of images containing a vast set of features/characters categorized under supervision. *In the absence of a large dataset, one approach is artificially enriching the data using interpolation techniques for generating missing time frames and densify the feature-based information content.*

*1) Interpolation:* Interpolation might increase the probability to distinguish successive peaks in the frequency domain. The probability can be controlled using an apt method of interpolation. It is expected that this step will enhance the interpretability of characters for the DL algorithm. It has been used for enhancing the image quality [19], [20]. Another technique involves optical flow-based temporal interpolation by using backward warping [8], [9]. Optical flow based method is the preferred approach with an atmosphere full of clouds to obtain interpolation related image processing characters.

Estimated image velocity by an optical flow can be used for supervised scene interpretation to an unsupervised dynamic investigation. Many methods for computing optical flow have been proposed. For in-depth insight, we refer to Barron *et al.* [21]. The process of determining optical flow is generally carried out through utilizing a brightness constancy constraint equation (BCCE). The relation makes the use of spatio–temporal derivatives of image intensity [22], [23]. Determining optical flow using the BCCE is an ill-posed problem. Classical gradient based methods, for example, Cauchy's method [24], [25], Newton's method [26], Marquardt's method [27], conjugate gradient method [28], quasi-Newton methods [26], [29], are useful solving underlying optimization problem.

Smoothness constraints by Horn and Schunck's method found helpful minimizing the distortions in optical flow estimation [30]. Combination of local and global methods was introduced by Bruhn *et al.* [31] to deal with ill-posedness. Spatial and temporal derivatives were used as constraints to overcome the ill-posedness issue. The choice of initial guess affects the performance of an iterative optimization technique resulting in a global or local extremum. Previously estimated optical flow fields were used as initial estimates by Giaccone and Jones [32]. A perceptually weighted optical flow was also proposed by Malo *et al.* [33]. Backward-warping [20] method for temporal interpolation was initially proposed by Ehrhardt *et al.* [8], [9]. This discussion inspires for carrying out a sensitivity analysis using various optimization methods and error estimates.

### C. Motivation and Methodology

In this article, the DL algorithm is used for the classification of satellite images under storm or nonstorm category. It is also used for locating the eye of the storm so that the regression model can be fitted for prediction automatically. The training process involves the formation of the system of equations, constraints, and its preferred solutions as input. These systems of equations and constraints can be solved by classical optimization techniques. There are several optimization methods and error estimates that can be used but the DL algorithm may not be sensitive to all. Thus, it is important to choose a proper combination of methods to use for a given dataset. The present work illustrates a case study for cyclone data. The manuscript presents an exhaustive study testing multiple mathematical frameworks (14) and several error estimates (6) for converting sparse into sufficient data enhancing its usability for the DL algorithm. Multiple satellite data having cyclones are tested and compared with previously reported estimations. Optical flow estimation methods are compared in terms of error after temporal interpolation. Generalized model to compute optical flow incorporating fractional order (FO) gradients is preferred over Brox's method and Horn and Schunck after testing. This model furnishes the dense optical flow while preserving the discontinuities at sharp boundaries [34], [35], good for images with clouds. In order to perform interpolation, inversion of an optical flow vector is required. A sensitivity analysis is performed for inverting the optical flow vector using 14 classical optimization methods. Performance metrics, namely: 1) mean square error (MSE), 2) mean difference error (MDE), 3) number of sites of disagreements (NSDs), 4) percentage error (PE), 5) peak signal-to-noise ratio (PSNR), and 6) sharpness are used for choosing the best solution method. Again DL platform and the artificially enriched data carrying storm features are used for classifying the cyclonic weather and estimation of the location of vortex of the cyclone. Finally, the regression method is used for predicting the path of cyclone. Th same is illustrated in Fig. 1 in the form of flow diagram. Satellite data available at NASA and ISRO servers are used for training and testing purposes.

## II. Methodology

This section is divided into two parts: 1) temporal interpolation algorithms and 2) DL frameworks. The interpolation section explains mathematical formulation for estimation of optical flow and algorithm.

### A. Optical Flow

We present an improved version of Horn and Schunck's method [22] for computing the optical flow field. A detailed version can be found elsewhere [34]; here, a summary is presented. The energy functional [22] is given as follows:

$$E_{\text{data}}(p) = \int_{\Omega} \left[ \left( \nabla I^T p + I_t \right)^2 + \alpha^2 \left( |\nabla u|^2 + |\nabla v|^2 \right) \right] dxdy \tag{1}$$

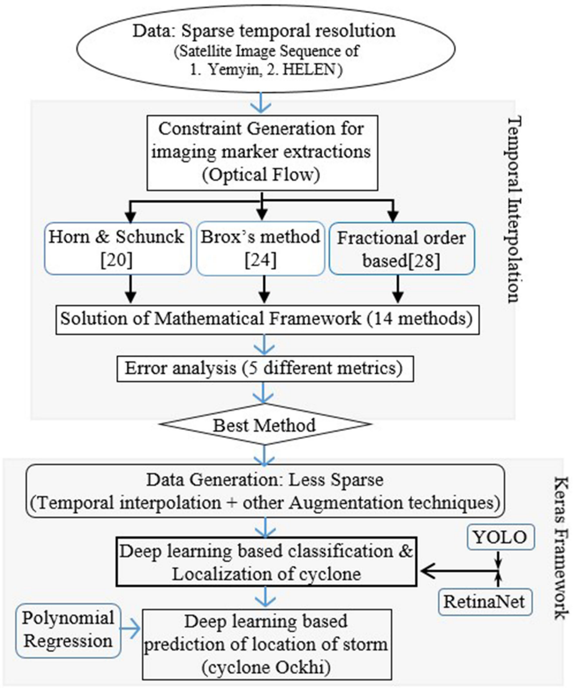
Data: Sparse temporal resolution
(Satellite Image Sequence of
1. Yemyin, 2. HELEN)
Constraint Generation for imaging marker extractions (Optical Flow)
Horn & Schunck [20]
Brox's method [24]
Fractional order based[28]
Solution of Mathematical Framework (14 methods)
Error analysis (5 different metrics)
Temporal Interpolation
Best Method
Data Generation: Less Sparse
(Temporal interpolation + other Augmentation techniques)
YOLO
Deep learning based classification & Localization of cyclone
RetinaNet
Polynomial Regression
Deep learning based prediction of location of storm (cyclone Ockhi)
Keras Framework

Fig. 1. Flow diagram describing the methodology.

where $\nabla I$ is the gradient of image intensity, $I_t$ is the temporal derivative of image intensity, $P = (u, v)^T$ is the velocity vector (optical flow components), and $\alpha > 0$ is a regularizing parameter.

This model was combined with Nagel and Enkelmann [38] and the energy functional becomes [35]

$$E_{\text{data}}(p) = \int_{\Omega} \left[ \left(\nabla I^T p + I_t\right)^2 + \beta^2 \left(p^T p + \lambda (|\nabla p|)^2\right)\right] d\Omega. \tag{2}$$

Here, $\beta$ is a constant value. This model preserves the discontinuity and gives a dense optical flow. It was further improved by incorporating FO derivatives optical flow components [34]. The energy functional is given as

$$E_{\text{data}}(p) = \int_{\Omega} \left[ \left(\nabla I^T p + I_t\right)^2 + \beta^2 \left(p^T p + \lambda (|D^{\alpha} p|)^2\right)\right] d\Omega. \tag{3}$$

Here, $D^{\alpha} := (D_x^{\alpha}, D_y^{\alpha})^T$ is called the left fractional derivative operator of Riemann–Liouville and $|D^{\alpha} u|$ is defined as

$$|D^{\alpha} p| = \sqrt{\left((D_x^{\alpha} u)^2 + (D_y^{\alpha} u)^2\right)}.$$

This model is a generalization of the integer-order variational optical flow models. The variational functional, (3), can also be written as follows:

$$E_{\text{data}}(p) = \int_{\Omega} \left[ \left(\nabla I^T p + I_t\right)^2 + \beta^2 \left(u^2 + v^2\right) + \lambda \left(|D^{\alpha} u|^2 + |D^{\alpha} v|^2\right)\right] d\Omega. \tag{4}$$

A detailed description of the method can be found elsewhere [34]. We choose $\alpha = 0.8$, $\beta = 600$, and $\lambda = 1$. Detailed information about the choice of these values is given by Kumar *et al.* [34].

We will also be estimating the optical flow using Brox's method and Horn and Schunck. However, the technical details are not given here for the sake of brevity. Interested reader may follow [30] for more detail.

### B. *Temporal Interpolation Algorithms*

Ehrhardt *et al.* [8], [9] have proposed an interpolation technique that utilizes the weighted average of optical flow components. The interpolation equation is given as follows:

$$I(x,t) = (1-\delta t)\cdot I(\mathbf{x} - \delta t \cdot \mathbf{u}, t_i) + \delta t \cdot I\left(\mathbf{x} - (1-\delta t)\cdot \mathbf{u}^{-1}, t_{i+1}\right). \tag{5}$$

Here, *I* is the image intensity, **x** is the position vector, $t_i$ and $t_{i+1}$ are two consecutive time instances, $\mathbf{u} = (u, v)$ is the optical flow vector, $\delta t = t - t_i$ and time is normalized, i.e., $t_{i+1} - t_i = 1$. For computing $\mathbf{u}^{-1}$, Ehrhardt *et al.* [8], [9] have used Newton–Raphson method. Shakya and Kumar [39] have also used this method for interpolation with optical flow vectors computed from the FO-based method. We will be using various techniques for numerical interpolation of the above-mentioned formulae.

*Method 1:* Moore–Penrose pseudoinverse of the optical flow vector.
*Method 2:* Scattered data interpolation (*SDI*) for the inverse of optical flow vector.
*Method 3:* Nearest neighbor (*NN*) calculation for the inverse of optical flow vector.
*Method 4:* Thin plate spline (*TPS*) interpolation.
*Method 5:* Kernel regression (*KR*).
*Method 6:* Sigmoid function interpolation (*SFI*).

A brief description of the above-mentioned methods is given as follows.

*Method 1:* Moore–Penrose pseudoinverse of optical flow vector: A matrix $\mathbf{A}_{m\times n}$, with $m = n$ or $m \neq n$, can be decomposed using singular valued decomposition into one diagonal matrix $\Sigma$ and two orthogonal matrices $\mathbf{U}$ and $\mathbf{V}$ such that $\mathbf{A} = \mathbf{U\Sigma V}^T$. The pseudoinverse of the matrix is defined as $\mathbf{A}^+ = \mathbf{U\Sigma}^+\mathbf{V}^T$. This method is used to compute the inverse of the optical flow vector and interpolation has been done using (5).

*Method 2: SDI* for the inverse of the optical flow vector: SDI is another technique that has been used to compute the inverse of optical flow vector. Following Shakya and Kumar [39], if we define a forward transformation **T** from source image space **s** to target image space **t**, the points in source space are specified as **s** = **t** + **T(t)**.

Then, the inverse transformation is given as

$$\mathbf{I}(\mathbf{s}) = \frac{\sum_i w(d_i)\cdot \mathbf{I}(\mathbf{t_i} + \mathbf{T})}{\sum_i w(d_i)} \tag{6}$$

where

$$w\left(d_i\right) = \begin{cases} \left(\frac{1}{d_i} - \frac{1}{R}\right)^2, & \text{if } d_i \leq R \\ 0, & \text{otherwise} \end{cases} \tag{7}$$

is the distance weight function associated with each interpolation point, $d_i$ is the distance of interpolated point from the $i$th data point, and *R* is the search radius.

*Method 3:* NN calculation for the inverse of the optical flow vector: Negative nearest forward transformation is used as inverse transformation to estimate the inverse of optical flow vector. If the nearest forward transformed point lies outside the source voxel, then averaging of surrounding voxels of forward transformed points is used.

*Method 4:* TPS interpolation: Having a set of $k$ points $P_j(x_j, y_j)$ and heights $h_j$ for $j = 1,\ 2, \ldots,\ k$, TPS interpolation is defined as [41]

$$f\left(x, y\right) = a_0 + a_x x + a_y y + \sum_{j=1}^{k} c_j U\left(x - x_j,\ y - y_j\right). \tag{8}$$

Here, constants $c_1, c_2, \ldots, c_k, a_0, a_x, a_y$ need to be find with property $f(x_j, y_j) = h_j \forall j = 1, 2, \ldots, k$. The term $U(x, y)$ is defined as

$$U\left(x, y\right) = \left(x^2 + y^2\right) \ln\left(x^2 + y^2\right).$$

*Method 5. KR:* Following Mühlenstädt and Kuhnt [42], consider a set $\mathbf{x} = \{x_1, x_2, \ldots, x_n\}$ with $N$ simplices $S_j = 1, 2, \ldots, N$ and vertices $x_0^j, x_1^j, \ldots, x_k^j$. $T$ is the Delaunay triangulation of the set $\mathbf{x}$. A linear function $\hat{y}_j(x) = \beta_0^j + x^T \beta^j$ for all $S_j$ can be fitted that will interpolate the vertices. The *KR* interpolator is constructed on polygons and it is combined with locally fitted linear functions [42]

$$\hat{y}_j\left(x\right) := \begin{cases} y_i & x = x_i, i = 1, 2, \ldots, n \\ \frac{\sum_{j=1}^{N} g_j(x) \hat{y}_j(x)}{\sum_{j'=1}^{N} g_{j'}(x)} & \text{elsewhere} \end{cases}. \tag{9}$$

*Method 6. SFI:* We used a sigmoid function for interpolation. For univariate logistic curve, it is defined as

$$S\left(x\right) = \frac{1}{1 + e^{-\beta x}} \tag{10}$$

where $\beta$ is the logistic growth rate or steepness of the logistic curve.

The above-mentioned methods are categorized as direct interpolation methods. We are also incorporating iterations on some of those techniques. Solution is updated, locally and globally, after each iteration. The local method inverts point by point in target space, whereas the global method inverts the whole field in source space [40]. Therefore, results of interpolation with 14 methods: 1) pseudoinverse, 2) pseudoinverse with local convergence, 3) pseudoinverse with global convergence, 4) *SDI*, 5) *SDI* with local convergence, 6) *SDI* with global convergence, 7) *NN*, 8) *NN* with local convergence, 9) *NN* with global convergence, 10) random with local convergence and 11) random with global convergence, 12) TPS interpolation, 13) *KR*, and 14) *SFI* are compared (see Table III). Here, an initial choice of optical flow vector is chosen randomly between maximum and minimum values of optical flow components.

We use the following metrics for comparing the results from different interpolation techniques.

1) *MDE:* Let $I_\tau^{\text{intp}}(x)$ and $I_\tau^{\text{orig}}(x)$ represent interpolated and original image intensities at pixel position $x$ in the $\tau$th frame. $N_\tau$ denotes the number of interpolated images and $\Omega_\tau$ the set of pixels in frame $\tau$. Then, the MDE is defined as [8]

$$\text{MDE} = \frac{1}{N_\tau} \sum_{\tau=1}^{N_\tau} \frac{1}{\left|\Omega_\tau\right|} \sum_{\mathbf{x} \in \Omega_\tau} \left| I_\tau^{\text{intp}}\left(\mathbf{x}\right) - I_\tau^{\text{orig}}\left(\mathbf{x}\right) \right|. \tag{11}$$

2) *NSD:* This metric is defined by the number of pixels where the difference between $I_\tau^{\text{intp}}(x)$ and $I_\tau^{\text{orig}}(x)$ is greater than a threshold value $\Theta$. It is defined as follows [8], [9]:

$$\text{NSD} = \sum_{\tau=1}^{N_\tau} \sum_{\mathbf{x} \in \Omega_\tau} \delta \left| I_\tau^{\text{intp}}\left(\mathbf{x}\right) - I_\tau^{\text{orig}}\left(\mathbf{x}\right) \right| \tag{12}$$

where

$$\delta\left(z\right) = \begin{cases} \theta, & \text{if } z < 0 \\ 1, & \text{otherwise} \end{cases}. \tag{13}$$

Here, $\theta$ is threshold and it is chosen 5% in the present study.

3) *PE:* It is defined as follows [43]:

$$\text{PE} = \frac{\text{Avg Error}}{\frac{1}{N} \frac{1}{M} \sum \sum \left[u_{\text{orig}}^2 + v_{\text{orig}}^2\right]^{1/2}}. \tag{14}$$

Here, Avg Error is the average error defined as [43]

$$\text{Avg Error} = \left[\sum \sum \left[\frac{D1_e^2 + D2_e^2}{N_e M_e}\right]^{1/2}\right] \tag{15}$$

where $D1_e = u_{\text{orig}} - u_{\text{int}}$ and $D2_e = v_{\text{orig}} - v_{\text{int}}$. Moreover, *u* and *v* are horizontal and vertical components of flow, respectively. Suffixes *orig* and *int* are used for original and interpolated flow. (*N*, *M*) is the size of ground truth image. Term $(N_e,\ M_e)$ is the size of $D1_e$.

4) *PSNR:* A PSNR value is calculated between true and estimated flow [43]

$$\text{PSNR} = 10\log_{10} \frac{255^2}{\text{MSE}} \tag{16}$$

where *MSE* is the mean squared error

$$\text{MSE} = \frac{1}{M \times N} \left[\sum_{i=0}^{M} \sum_{j=0}^{N} \left(I^{\text{int}}\left(i, j\right) - I^{\text{orig}}\left(i, j\right)\right)\right]^2.$$

5) Sharpness ratio: This metric is defined as follows:

$$\text{sharpness} = \frac{\sum \left\{\left(df/dx\right)^2 + \left(df/dy\right)^2\right\}}{N}. \tag{17}$$

### C. DL Frameworks

For classification purposes, a basic Keras model [44] was trained on the dataset. The platform provided by Google colaboratory [45] was used for training. Standard convolutional neural network (CNN) models like Xception [46], NasNetMobile, and MobileNet were also applied on the preprocessed dataset using Keras applications.

The methodology for object detection and localization is to break down the image into multiple segments and feed each segment to the model and get the label. It is highly likely that the object will appear half cropped in one segment. In order to get complete object in our segment, we need to work with images of different scales. This increases the computational cost and the inference time, thus making the model impractical in real-time scenarios. This idea, implemented in a sliding window technique, is fairly outdated [47].

To overcome the cost of time as well computation, you only look once (YOLO) [37] is preferred in this work, for detecting the circular rotating area. YOLO system helps in detection of objects in real time. It consists of 23 convolutional layers and it uses batch normalization technique and leaky ReLU activation [37]. Unlike sliding window technique or regional proposal network, YOLO considers the whole image and thus encodes the contextual information about the classes and their appearance. Fast region-based CNN (R-CNN) [48], a top detection method, mistakes the background patches as an object because it cannot see the larger context. The number of background errors is significantly reduced with YOLO in comparison to fast R-CNN [47]. The YOLO pixelates input image into $N \times N$ grid cells. A particular grid detects an object, if its center falls into that grid cell. Each grid cell predicts $B$ number of bounding boxes and confidence scores for those boxes. The confidence score tells about the accuracy of prediction. For each bounding box, five values $x$, $y$, $w$, $h$, and confidence score are predicted, where $(x, y)$ represents the center of box relative to the grid cell. The width $(w)$ and height $(h)$ are predicted relative to the whole image.

In YOLO, an individual grid cell is allowed to contain a single class with the capability to predict two bounding boxes only. This spatial constraint limits the prediction capability of the number of nearby objects. Overall, the model struggles with the detection of small objects that appear in groups. RetinaNet [49], a relatively sophisticated model, uses Resnet and feature pyramid network. Single-stage detectors are less accurate but had fast inference time, whereas two-stage detectors are more accurate but took significant time during inference. RetinaNet is a modified model of a single-stage detector with improved accuracy by modifying the loss functions. It outperforms the two-stage/shot detector faster R-CNN in terms of speed as well as accuracy. The cross-entropy/focal loss (*FL*) function (18) is reshaped by adding a modulating factor $(1 - p_t)$ to downweight low level examples and focusing parameter $\gamma$. The constraint is meant to training on hard negatives

$$\mathrm{FL}(p_t)^{\gamma}\log(p_t) \qquad (18)$$

where focusing parameter $\gamma$ is tested in the range of [0, 5] in the experiment. $\gamma = 2$ works best in our experiment. The focusing parameter smoothly adjusts the rate at which easy examples are downweighted. It affects the function of modulating factor [49]. If an example is misclassified and model's estimated probability $p_t$ is small, the modulating factor is near 1 and the loss is unaffected. For high probability values, the factor becomes to 0 and the loss for well-classified examples is downweighted.

TABLE I
RAW DATA EXTRACTED FROM THE IMD SITE FOR INTERPOLATION

| **Cyclone Name** | **Year** | **Peak Category** | **No. of Images** | **Image Type** |
|---|---|---|---|---|
| AILA | 2009 | Severe Cyclonic Storm | 190 | Visual |
| BOB3 | 2007 | Cyclonic Storm | 129 | Visual |
| GONU | 2007 | Super Cyclonic Storm | 112 | Infrared |
| LEHAR | 2013 | Very Severe Cyclonic Storm | 111 | Infrared |
| PHAILIN | 2013 | Very Severe Cyclonic Storm | 103 | Infrared |
| PHYAN | 2009 | Cyclonic Storm | 141 | Visual |
| RASHMI | 2008 | Cyclonic Storm | 170 | Infrared |
| FANOOS | 2005 | Severe Cyclonic Storm | 39 | Visual |

## III. EXPERIMENTAL RESULTS

### A. Dataset

For interpolation, we used the satellite images obtained from KALPANA-I. Image data are downloaded from the India Meteorological Department (IMD) archive [50] for the cyclone in June 2007. In particular, we processed (Yemyin) the images for June 21, 2007. A depression area was declared by IMD near east-southeast of Kakinada, Andhra Pradesh, India. For DL classification, we downloaded data from IMD archives [50] that contain images of cyclones from the year 1990 until recent times with increased accuracy and coverage in recent years. With the advancement in technology infrared, midinfrared, short-wavelength infrared, water vapor images of the recent cyclones were also included in the archive. Raw data are selected for the analysis given in Table I. For prediction, we trained the model with several random cyclone images downloaded from Internet. Also, we downloaded images from Meteorological and Oceanographic Satellite Data Archival Center (MOSDAC) [51]. We tested the model on Ockhi cyclone that occurred on December 2, 2017. Python codes and libraries are used for creating the labels. The resolution of each image is $1024 * 1200$ pixels.

### B. Interpolation Results

Temporal interpolation plays an important role when the image data is available at large time intervals. We are processing images of cyclone Yemyin, which happened in June 2007. Images obtained from satellite KALPANA-I are captured at 1 h of interval. We first used two images of 2 h of interval (05:00:03 hours and 07:00:03 hours (GMT) on June 21, 2007) and generated the image at 1 h (06:00:03 hours) of interval. The ground truth image is also available to compare. In order to get the intermediate images, we used (13). The optical flow is estimated with three approaches: Brox's [12] method, Horn and Schunck [22], and

TABLE II
COMPARISON OF INTERPOLATION RESULTS WITH VARIOUS METHODS USING BROX [11] METHOD FOR COMPUTING OPTICAL FLOW

| S.N. | **Method** | **Time** | **Iterations** | ***MDE*** | ***NSD*** | ***PE*** | ***PSNR*** | **Sharpness** |
|---|---|---|---|---|---|---|---|---|
| 1 | *Pinv* | 15.03 | 1 | 9.78 | 78340 | 1.92 | 46.46 | 0.89 |
| 2 | *Pinv* + Global Iteration | 68.76 | 6 | 3.94 | 59784 | 0.70 | 47.54 | 0.96 |
| 3 | *Pinv* + Local Iteration | 15.94 | 40 | 3.95 | 59729 | 0.69 | 47.59 | 0.96 |
| 4 | *SDI* | 2.97 | 1 | 3.94 | 59752 | 0.69 | 47.56 | 0.96 |
| 5 | *SDI* + Global Iteration | 256.57 | 20 | 3.94 | 59726 | 0.69 | 47.59 | 0.96 |
| 6 | *SDI* + Local Iteration | 16.64 | 40 | 3.95 | 59738 | 0.69 | 47.59 | 0.96 |
| 7 | *NN* | 2.4 | 1 | 3.94 | 59715 | 0.69 | 47.58 | 0.96 |
| 8 | *NN* + Global Iteration | 229.68 | 20 | 3.94 | 59726 | 0.69 | 47.59 | 0.96 |
| 9 | *NN* + Local Iteration | **3.01** | 40 | 3.94 | 59721 | 0.69 | 47.59 | 0.96 |
| 10 | *Random* + Global iteration | 235.85 | 20 | 3.90 | 59101 | 0.72 | 47.21 | 0.97 |
| 11 | *Random* + Local iteration | 0.71 | 40 | 3.95 | 59866 | 0.69 | 47.64 | 0.96 |
| 12 | *TPS* | 689 | 1 | 2.94 | 48871 | 0.46 | 50.91 | 0.96 |
| 13 | *KR* | 121 | 1 | 2.79 | 43075 | 0.61 | 48.60 | 0.95 |
| 14 | *SFI* | 62.41 | 1 | 2.33 | 35553 | 0.60 | 48.87 | 0.94 |

TABLE III
COMPARISON OF INTERPOLATION RESULTS WITH VARIOUS METHODS USING FO OPTICAL METHOD [15] FOR COMPUTING OPTICAL FLOW

| **Method** | ***MDE*** | ***NSD*** | ***PE*** | ***PSNR*** | ***Sharpness*** |
|---|---|---|---|---|---|
| *Pinv* | 8.48 | 65620 | 0.19 | 63.95 | 0.94 |
| *TPS* | 2.94 | 48871 | 0.63 | 55.20 | 1.02 |
| *KR* | 2.79 | 43075 | 0.71 | 54.15 | 0.99 |
| *SFI* | 2.34 | 35553 | 0.65 | 54.86 | 1.04 |
| **Other** | **3.73** | **56194** | **0.09** | **72.25** | **0.98** |

FO derivatives [43] based method. It is found that Brox's method and FO derivation method provide results under acceptable error estimation for considered data. Results by Horn and Schunck are omitted here due to brevity. Results are compared in Tables II and III, respectively. A comparative study is made based on different interpolation methods mentioned above [*Methods 1–6*] with global and local convergences. Error is computed with image intensities and optical flow vector. *MDE* is one of the error estimates that computes the differences of image intensities between the original and the interpolated image. Looking into both Tables II and III, the *MDE* values for different interpolation methods, *SFI* has the lowest values of 2.33 with Brox's method and 2.34 with the FO-based method. It indicates that performance of *SFI* is comparatively better than other methods, if MDE is the criterion. Different metric values are comparatively lower for the *FO*-based method. The next error measure is *NSD*, which is the number of pixels where the difference of image intensities between original and interpolated is greater than 0.05. Thus, the smaller number of *NSD* indicates close similarity to the original image. This error estimator's behavior is consistent with the *MDE* values, i.e., *SFI* giving the minimum NSDs. Another error estimate is *PE*, which is computed from the difference of optical flow components estimated from the original image and the interpolated image. The behavior of *PE* is different from what we observed with the previous two error measures, which were based on image intensity differences. A minimum value of 0.46 is observed with the *TPS* interpolation method while using Brox's method for computing optical flow. However, we observed a very small value of 0.09 with other methods ($P_{\text{inv}}+$ local and global iterations, *SDI*, *SDI* + local and global iterations, *NN*, *NN* + local + global iterations, random + local and global iterations) with optical flow components estimation from FO derivative based method. *PSNR* value is also best for these interpolation methods. From overall observation, we preferred to use the *NN method with local convergence* due to less computational cost, fairly comparative error estimates, and sufficient iterations before convergence aspects.

Next, we generated 14 images between the images captured at 05:00:03 hours and 07:00:03 hours. The interpolated images are shown in Fig. 2(a1)–(a13). Leftmost image in the first row and rightmost image in the fourth row [see Fig. 2(a) and (b)] are the original images captured at 05:00:03 and 07:00:03 hours, respectively. Images are interpolated at approximately 8 min of interval; the first row contains images from 05:00:03 hours to 05:24:03 hours (left to right). The second row displays the images from 05:32:03 hours to 05:56:03 hours (left to right). The third row contains images from 06:04:03 hours to 06:28:03 hours (left to right). The last row has images from 06:36:03 hours to 07:00:03 hours (left to right). By closely observing, we found that the second image in the first row is close to the image captured at 05:00:03 hours and the second last image in last row is much more similar to the image at 07:00:03 hours. Ground truth images are not available for small temporal steps (8 min). Four instances are marked with red boundaries around cyclone boundary. It can be observed that artificially interpolated images show a smooth transformation of the cyclone (circular shape and relatively small diameter) at 05:00:03 hours into cyclone depicted into second image (relatively bigger diameter and oval shaped). These missing (during the measurement) but artificially generated images are expected to help the neural network to classify storm. Similarity check between interpolated and original images is performed using shape comparison algorithm [52]. Relative change (w.r.t. first recorded image) in the Hausdorff dimension decreases with the time series of interpolated image. It is shown in Fig. 2(c).

## C. Deep Learning

We applied DL techniques for two purposes: 1) for classifying an image under category being part of storm or not containing

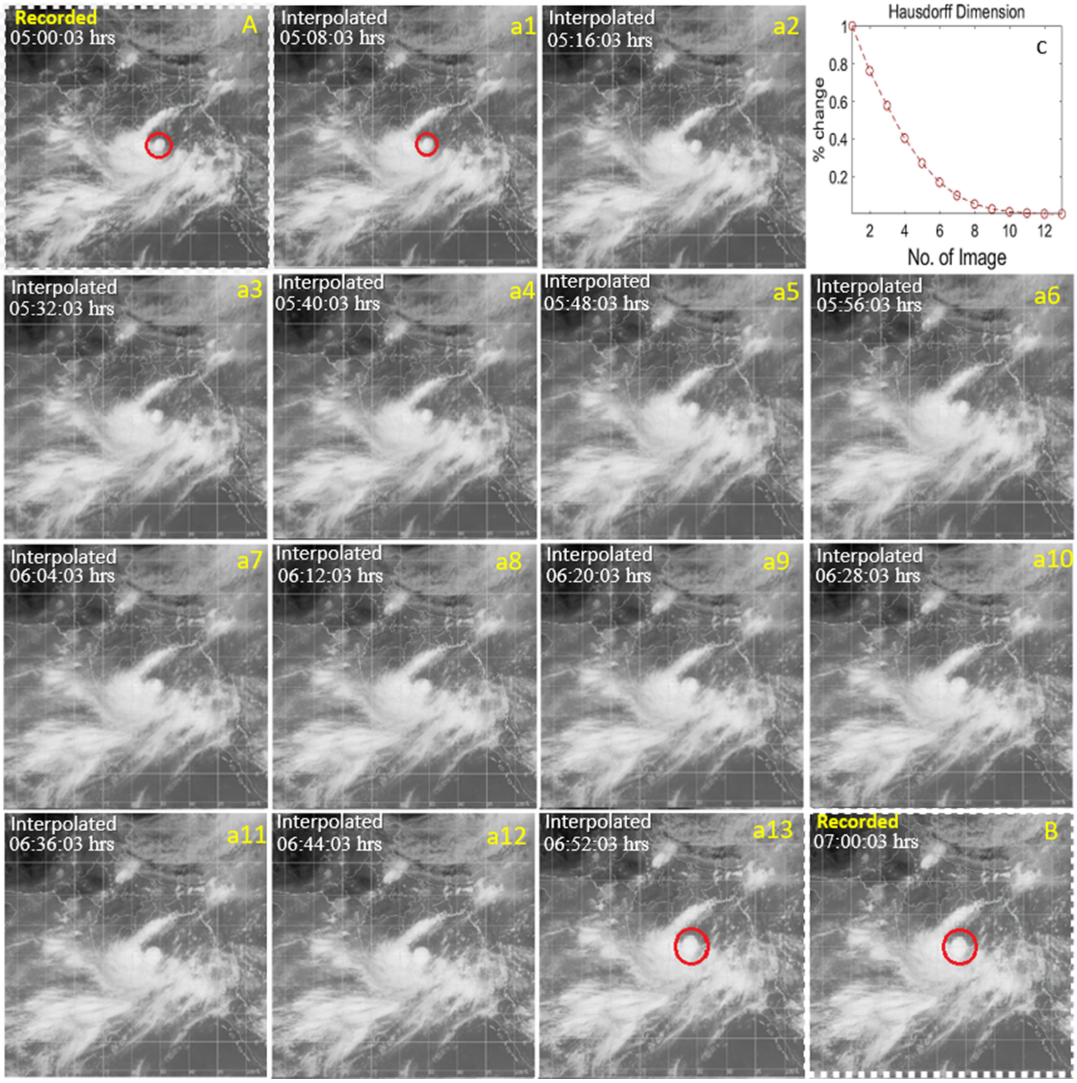

Fig. 2. (a) Top left and (b) bottom right images are original (recorded via satellite: movie A) images captured on June 21, 2007 at 05:00:03 hours and 07:00:03 hours (GMT), respectively. From the second image (2a1) in first row to the second last image (2a13) in fourth row are the interpolated images. In total, 14 images are generated between 05:00:03 hours to 07:00:03 hours at equal time interval. Interpolated images translate characters well in-between both of the recorded images. (c) Hausdorff dimension for each image.

any characters of the existence of storm and 2) for predicting the storm location in near future.

*1) Classification of Storm and Nonstorm Weather Conditions:* Tropical cyclones are a regular phenomenon in the North Indian Ocean, which affect the Indian subcontinent mainly from May until mid-December causing significant loss of lives and damage to property. The region specialized meteorological center for tropical cyclone over the North Indian Ocean, IMD, keeps a track of the North Indian Ocean cyclones and their trajectories, and issues a four-stage warning on any cyclonic weather system. The four stages include a *precyclone, precyclone watch, cyclone alert, cyclone warning and post-landfall lookout*.

These four stages of tracking and warning issuance are determined by the different stages of the development of a cyclonic storm. According to IMD's classification, North Indian Ocean cyclone generally starts out as a depression with a wind speed of 31–50 km/h over the Bay of Bengal or the Arabian Sea. Depression intensifies into a deep depression when wind speed reaches 51–62 km/h and the system starts drawing in more moisture. When the wind speeds further intensify to 63–88 Km/h with longer sustenance, the IMD classifies it as a cyclonic storm and assigns it a name. The next stages of the severe cyclonic storm are reached when the wind speeds peak to a range of 118–165 km/h and have a potential of huge damage to life and property. On further intensification, the cyclone is subsequently classified into a very severe, extremely severe, and super cyclonic system depending on the wind speeds. Weather prediction and sensitivity analysis of these storms have a profound sociological and humanitarian value. The advancements in RS, weather prediction research, and evacuation procedures in the last decade have significantly decreased the damage to life caused by these cyclones. The sensitivity and accuracy analysis of these predictions are very important in this domain. A false alarm can cause extraneous expenditure in mobilizing and

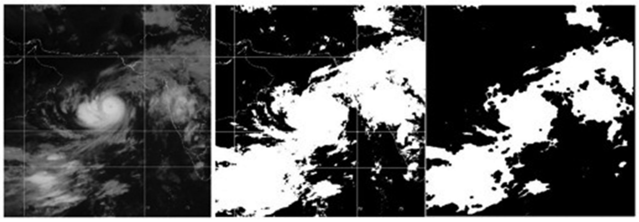

Fig. 3. Leftmost image is the original image, middle image after binarization, and the rightmost image after erosion.

evacuating people and a delayed response to weather intensification can amplify the damage incurred.

DL has shown encouraging performance for the characterization of complex patterns such has storm classification [51]. We deploy the CNN framework using Keras (open source neural network library) for the classification of storm images into the five categories specified by IMD. The satellite images (infrared and visual) of eight cyclones from IMD are fed as input to the CNN and the model is trained on Google collaboratory platform to extract relevant features from these images. Temporal interpolation of images using optical flow based interpolation [8], [9] is used to augment the data. Results are highly promising and contain great future scope. We first preprocessed the images to remove the unnecessary information, which could make the model computationally costly.

Image preprocessing: Before training the data, we preprocessed the images. This step removes the unwanted and noisy features and also increases the efficiency of the model. The preprocessing steps mentioned in [53] are used as a skeleton with some changes, which were optimal for our dataset.

Raw image from the dataset was cropped to remove header files and white edges. Subsequently, image binarization with some additional changes was applied to the images. Objective of this step was to remove the unnecessary information like grid lines, geographical boundaries, and landscape. In classic image binarization, the image is converted into a binary image with pixel intensities above a certain threshold are converted to one and others to zero. In our algorithm, the output was still an RGB image and the pixels with intensities above a threshold retained their original values while the other pixels were converted to the lowest minimum, i.e., zero. As a result, only the vortex and peripheral cloud patches were retained with original pixel values and other features were removed. The threshold was taken to be suitably optimized multiple (generally 1.4) of median of the pixel intensities of the image. Subsequently, image erosion was applied on the processed image. The mathematical definition of image erosion is presented in [53]. Fig. 3 shows the preprocessed steps for a satellite image. The unnecessary features like grid lines and landscapes are filtered also.

Model training and output: A total of 995 images were downloaded from the IMD archive [50] but for training of our model, data augmentation was required. Optical flow-based temporal interpolation was applied to the images and ten images were obtained between two temporally consecutive images.

A basic Keras model was trained on the preprocessed dataset. The platform provided by Google collaboratory was used for training. The model was trained on 6930 augmented images and validated on 2970 images randomly shuffled and splitted from our original database. The model used is depicted in Table IV.

TABLE IV
Sequential CNN Model Developed and Employed in Image Classification of RS Images

| S. No. | Type | Channel of Filters | Filter/Pool Size | Activation |
|---|---|---|---|---|
| 1 | Convolutional | 32 | 3,3 | ReLU |
| 2 | Convolutional | 32 | 3,3 | ReLU |
| 3 | Max pooling | - | 2,2 | - |
| 4 | Convolutional | 64 | 3,3 | ReLU |
| 5 | Convolutional | 64 | 3,3 | ReLU |
| 6 | Max pooling | - | 2,2 | - |
| 7 | Dense | 128 | - | ReLU |
| 8 | Dropout | 0.3 | - | - |
| 9 | Dense | 5 | - | Softmax |

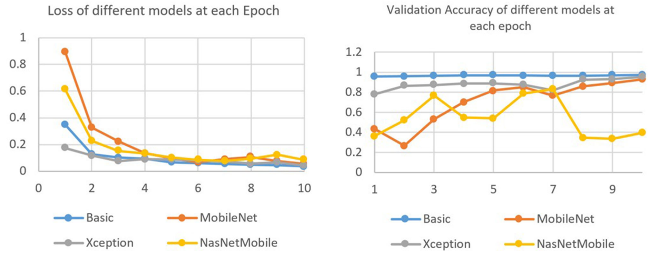

Fig. 4. Trend of loss and validation accuracy during the training of four different CNN models.

The above model was compiled with categorical cross-entropy loss and Adam optimizer [53]. A peak validation accuracy of 97% is achieved for classifying the storm or non-storm. Standard CNN models like Xception, NasNetMobile, and MobileNet were also applied on the preprocessed dataset using Keras applications.

Same train-test split was used on the standard models and the comparison of training loss, training accuracy, and validation accuracy of the four models is given in Fig. 4. The accuracy and loss (categorical cross-entropy loss) of the four models have been compared for ten epochs. When the accuracy of a model is high and the loss is low, then the model is believed to be trained well on the dataset. In the left graph, we can see that the Xception model starts out with the least loss but as the number of epochs increases, the basic model also reaches the same level. In terms of accuracy, the basic model has the highest accuracy from the start of the training and Xception catches up to it as the epochs increase. Both the basic model and Xception show highly positive results after ten epochs at the end of the training and can be used for further testing. For further work, the data can be augmented with other information like wind and temperature and a combined RNN and CNN model can be used for better classification of the storm.

To test the predicting efficiency of the model, cyclone Helen's data (not used in training of the model) are fed to the model. For validation, classification information of Helen is taken from previous work [54]. A peak validation accuracy of 45% is obtained when 120 images of Helen are fed into the model.

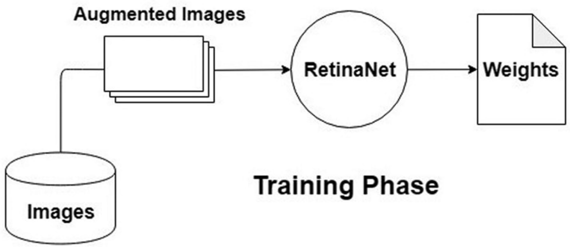

Fig. 5. Training phase.

Accuracy is less due to the inconsistency of Helen cyclone data. The model was then trained on six out of eight cyclones mentioned in Table I and validated on the other two (in these eight cyclone images inconsistencies have been removed by data augmentation and other processes). A peak validation score of 58% was achieved. An increase in accuracy highlights the power of data preprocessing.

### D. Prediction of Storm

Cyclone is a system of winds rotating about a center of low atmospheric pressure termed as "eye" having wind velocity above certain limit. Here, we are presenting a model to track the eye (or the center) of the cyclone. The aim of our model is to analyze the patterns present in its movement and then use it to predict the path of the cyclone. In order to achieve a robust model, we need to have a large amount of data. We did not have any official repository that managed the cyclone-annotated data. Therefore, we had to manually annotate the bounding boxes across the cyclone images. We took the cyclone images and annotated it using python library matplotlib [55]. Unfortunately, we were only able to extract hundreds of images but any DL network would easily over-fit on such a small dataset. To counter this we had to rely on augmentation. Robustness in incorporated in the modeling by artificially adding noise. We performed different augmentation techniques like scaling, translating, cropping, rotating, adding Gaussian noise, adding hue and saturation to it. After augmentation, we were able to create 5000 images for training and 500 images for testing. For testing, we used the archived cyclone images provided by the Indian Meteorological Department. These satellite images were captured over half an hour interval, over different spectrum (visible, infrared). We applied interpolation to generate intermediate images to provide large number of images from a particular cyclone.

Our model consists of the following two phases.

1) Training phase (see Fig. 5).
2) Prediction phase (see Fig. 6).

The DL model RetinaNet required a CSV file that contains the following format.

path2image, $x1$, $y1$, $x2$, $y2$, *obj,* where path2image represents the complete path of the image, whereas $(x1, y1)$ and $(x2, y2)$ correspond to the top-left coordinate and bottom-right coordinate of bounding box, respectively. As our goal was to only detect a cyclone, the *obj* was set to cyclone.

Prediction phase: We had to provide two CSV files: one for training and other for validation. The training was done over an NVIDIA GTX 1080 graphics card taking 24 h of processing time. We saved the weights of our network, so that it can be used later without training the network from the scratch.

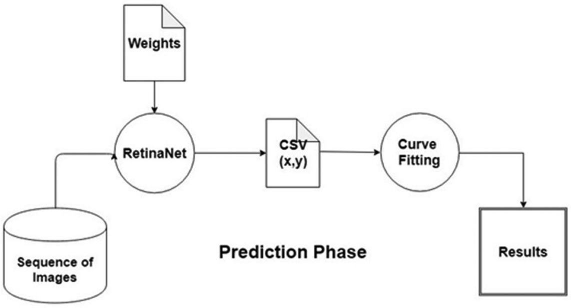

Fig. 6. Prediction phase.

In this phase, a sequence of images was fed to the RetinaNet, which was initialized with the weights that we have saved earlier. It produces the bounding boxes across the cyclones, i.e., it outputs ($x1$, $y1$) and ($x2$, $y2$) coordinates (see Fig. 7). As we are only interested in having the location of the centers, we calculated the center coordinates using the top-left and bottom-right coordinates and saved it to a CSV file. The path of the cyclone is highly temporal; thus, another DL-based model long short-term memory (LSTM) can be used to extract its temporal essence. But it has its own drawbacks. To train an LSTM, we need a high-frequency dataset that was not available [56]. Furthermore, some of the frames were missing from the videos, which then lead to a sudden change in center coordinates. Therefore, there was no homogeneity within the dataset, which forced us to find an alternative. RetinaNet is used. The model, which we used for training, takes care of class imbalance by modifying its loss and taking into account the examples that are less frequent in the dataset [49], [57].

The data of cyclone named Ockhi taken from MOSDAC [58] are used for showing the final analysis now. We have processed multiple cases studies/data from May 2016 to December 2017. Several instances with convincing cyclonic activities are observed in this duration. The case study from November 29, 2017 to December 5, 2017 shows the presence of a cyclone (termed as Ockhi). This cyclone was found transiting from Indian Ocean via Peninsular India toward Sea of Arabia. These data comprise 44 images for each day (each with 30 min time difference). Data on December 2, 2017 are discussed in Fig. 7. This image set is densified into 95 images using interpolation technique, each image with 15-min time interval. Fig. 7(a1)–(a8) shows images from whole day. Fig. 7(b1) contains overlay of images of Fig. 7(a1) and (a8). Further, Fig. 7(c1) is an ×5 zoomed section showing existence of two eyes of cyclone or vortex by $V_{c_1}$ and $V_{c_2}$. It also shows the path of the eye of cyclone for whole day, estimated by the DL algorithm and traced manually. Fig. 7(d1) shows separate trajectories (in red color: bigger circles by DL), which is predicted by both methods.

The DL algorithm estimated coordinates of 95 eyes, out of which 61 (with unique coordinates) are used for polynomial regression fitting, toward both ends. Coordinates (71 unique) of

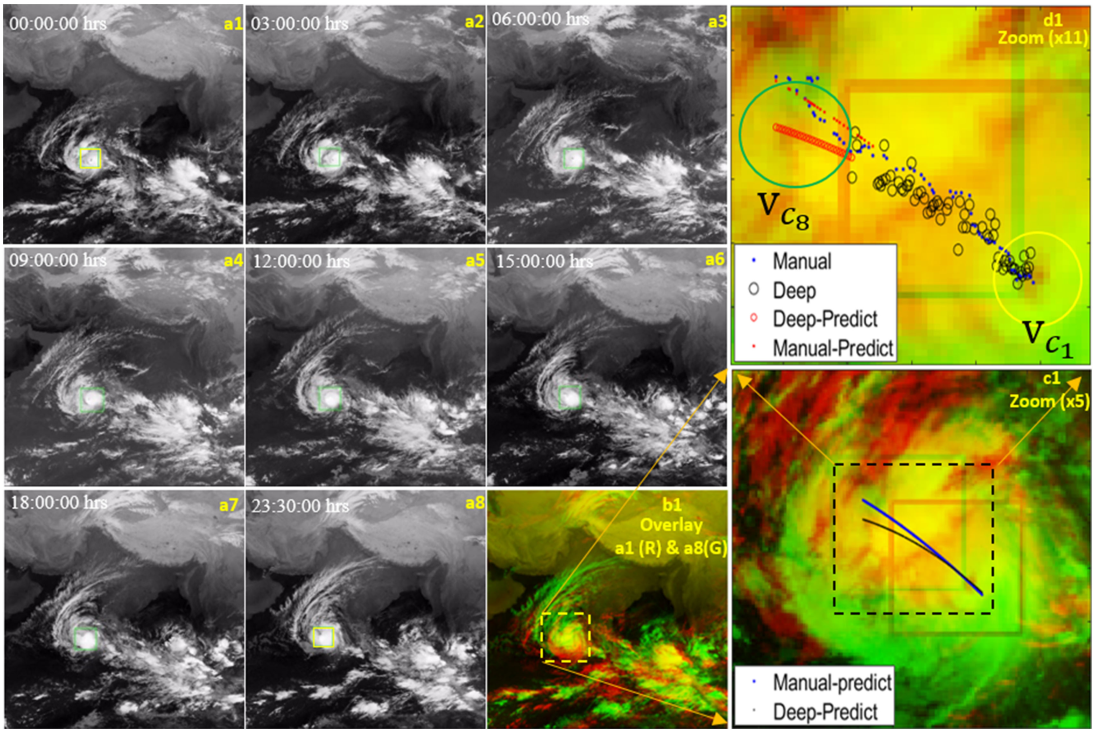

Fig. 7. Automatic locating the eye of vortex of a cyclone: (a1)–(a8) time series images (Movie 1) showing extracted boundaries of vortex between 00:00:00 hours and 23:30:00 hours; (b1) overlapped images a1 (red) and a8 (green); (c1) ×5 zoom part covering only the cyclone regions paths traversed by eye of the vortex (black: via DL algorithm, blue: via manually) is shown; (d1) ×11 zoom part shows coordinates of eye of vortex, initial and final vortex location $v_{c_1}$ and $v_{c_8}$, respectively.

eyes (see Fig. 7(d1) shown in black circles) are also located, manually. Six persons are given the data and apt training to visualize the eye of the vortex. The averaged data (see Fig. 7(d1) shown in blue dots) are fitted using same polynomial regression method for predicting the path. It is compared with the predicted path obtained by DL algorithm. Predicted path by DL algorithm appears to be originating nearby to the first eye [see Fig. 7(a1) and (d1)] and eye [see Fig. 7(a8) and (d8)] from the final image of the data. One can visualize clear presence of eye of cyclone in Fig. 7(a1)–(a4). Afterward in successive figures, these six persons had to estimate approximate centers. We note [see Fig. 7(a4)] that eye of cyclone not always remains at the center. This might be the reason of difference between manually and DL extracted path of cyclone in Fig. 7(c1). Average difference between fitted curve and original position (extracted manually) is found less than five pixels in every case that we analyzed. Our model correctly labeled it as no cyclone and outputs no bounding boxes, if cyclone activity was absent in images.

One pixel is equivalent to 1 km. The average velocity and maximum velocity of cyclone can be estimated by coordinates of eyes. In this particular case, these are found to be 30.1 and 140.2 km/h off course, a correlation is required for estimating the ground reality. Coordinates information with respect to time is used for this calculation. Close observation of the initial few images depicts appearance of two cyclones, one in left bottom and other, rather a week case, in right side. The weaker cyclone, on the right side, gets dissolved after 9 h. DL algorithm clearly avoids classifying it into storm case. We have found many instances (not this case study) with situations such as if in a time sequence multiple cyclones exist or will arise in the future.

TABLE V
DL Algorithm Estimations

| S.N. | Name | Duration | RMSE % |
|---|---|---|---|
| 1 | ROANU | 17-21 May 2016 | 15.55 |
| 2 | KYANT | 25-26 Oct. 2016 | 5 |
| 3 | VARDAH | 07-12 Dec. 2016 | 9 |
| 4 | MORA | 27-30 May 2017 | 6 |
| 5 | OCKHI | 29 Nov. – 06 Dec. 2017 | 11 |

In some instances cyclone with strong appearances dies out while weaker gets converted into strong cyclone. All such cases are tested, successfully. The algorithm is also tested with the option when images contain multiple cyclones together. In this figure that option is not used. The processed data between May 17, 2016 and December 6, 2017 are shown in Fig. 8. Table V contains the details of five different cyclones developed in this duration. Respective data are obtained from the Joint Typhoon Warning Center (JTWC) and shown in Fig. 8(a) [59]–[62]. JTWC data tracks are obtained from web repositories of 1) NCAR Lab, UCAR (yellow pin shape: only in case of Ockhi) and 2) North Indian Ocean Best Track Data by Naval Meteorology and Oceanography Command of U.S. Navy (multicolored cyclone shape). Coordinates of eyes of all five cyclones are obtained manually and shown in Fig. 8(b) and (c) using red dots [best seen in the composite image in Fig. 8(d)]. Fig. 8(c)

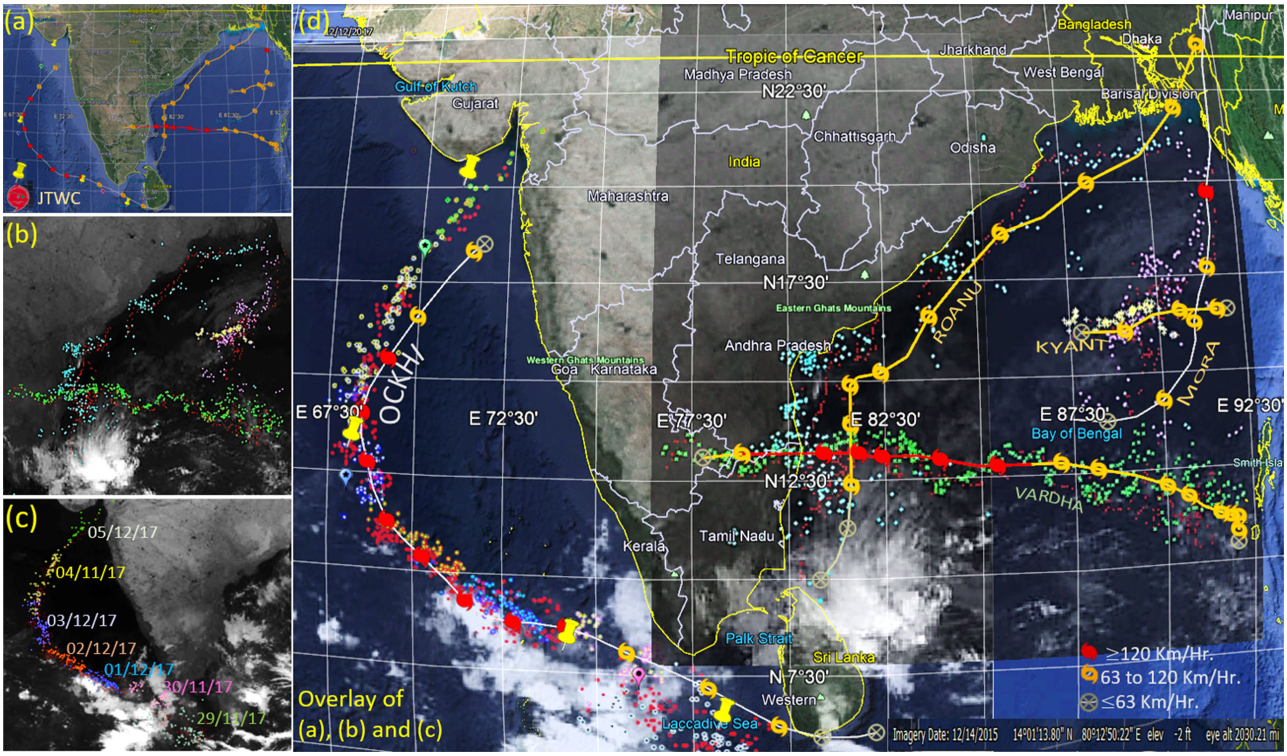

Fig. 8. Tracing the cyclone path. (a) Cyclones between 2016 and 2017 by JTWC records over Indian subcontinent with traced paths. (b) Location coordinates of cyclone in year 2016 using DL algorithm. (c) Location coordinates of cyclone in year 2014 using DL algorithm. (d) Overlay of coordinates of path of cyclone.

contains data related to Ockhi in Arabian Sea side of Indian Peninsula. The other four storms were present in Bay of Bengal side. Root-mean-square error (RMSE) is estimated between coordinates obtained by DL and coordinates obtained manually. Manually obtained coordinates are assumed near to be the true value. The DL algorithm has successfully detected instances of cyclonic activities in two-year durations in respective locations that match with manually obtained with less than 16% error.

Weaker instances (wind speed less than 55 km/h) namely, TWO, NADA, MAARUTHA, FOUR are skipped to keep the clarity in this figure. Fig. 8(c) shows datewise color-tagged estimated path (coordinates of eye of cyclone) using DL algorithm (green, pink, blue, yellow-red, blue, yellow, green, respectively, from November 29 to December 4, 2017). Fig. 8(d) is made by overlaying the images shown in Fig. 8(a)–(c).

For Ockhi, estimation by the DL algorithm, reported earlier and manually, is closer to each other from November 30, 2017 to December 4, 2017. For VARDHA (green dots by DL and curve with cyclone marker by JTWC), both match even better. It is observed that DL estimation and manual estimation are closer to each other but JTWC tracks digress when storm is about to weaken. The slope of the latter part of Mora and Ockhi slightly differs. JTWC tracks, otherwise, show similar characters with minor offset. The slight difference may be due to difference in perception of eye of the vortex by a human user and DL algorithm. The processed flat images do not have $z$-axis perception; eye of a cyclone visually appears to be a hole or darker region surrounded by clouds rotating in spiral or circular fashion. Sometimes, the camera of the satellite is not right on the top of the eye. The eye aligned according to curvature of earth but satellite's camera is not. Images in such conditions give impression that there are clouds on the top of the eye hiding it. In such situations, the only way to locate an eye is to assume that it would be at the center of the cyclone, which may not be true. It is observed that DL algorithm performs better identifying eye in such conditions when compared with human user. We note that there is a difference in JTWC data by two resources for Ockhi. One is depicted using yellow color button and another by storm shaped markers. Movie 2 contains time series migration of Ockhi in detailed fashion.

## IV. Conclusion

Several datasets are used as training and testing set for the development of a DL algorithm. Several interpolation methods are also tested for enhancing the performance of the DL algorithm. The basic CNN model outperforms all the standard models in the classification regime of RS images. In particular, the YOLO model is suggested for detecting and locating the cyclone. R-CNN model is suggested for predicting the location of storm.

Pointwise conclusions are given as follows.

1) The FO-based approach is slightly better than Brox's approach for optical flow estimation for interpolation.
2) SFI optimization approaches are better as far as MDE and NSD error estimates are used, but for slight compromise of performance nearest neighborhood method is considerably faster.

3) If the high-frequency dataset is not available, then RetinaNet is a better model than LSTM.
4) Performance of DL algorithm improves with densified dataset using interpolation and data augmentation.
5) Tracking and predicting eye of cyclone is much accurate by DL algorithm as compared to manual process, if images do not contain more than one cyclone.

Finally, DL algorithms for classification and prediction of storm in near future are successfully tested. For classifying the storm or nonstorm, an accuracy of 97% and for detection of cyclone a confidence of greater than 84% are achieved. The DL algorithm contains two different neural networks, one for classification and another for locating the eye of cyclone, trained separately. The performance of classification is expected to affect the performance of locating the eye of the cyclone. The flat images fail to impart effect due to curvature of the earth and need to incorporated separately in the future, if made available.

The outcome of the presented work depicts involved mathematical nature of issues and highlights an approach to find an optimal classical preprocessing candidate before employing neural network, in the form of a complete DL algorithm. *We note that advanced post processing DL algorithm such as presented in this article can help to predict the cyclone pathway for early preparedness without much human intervention with more accuracy and speed.*

## Acknowledgment

The authors gratefully acknowledge the Indian Space Research Organization, MOSDAC, and other similar open-access repositories for providing data. The authors acknowledge the early stage data processing contribution made by K. Kumar and SPARK Intern A. Agarawal. The authors would also like to acknowledge SPARK Intern Program Indian Institute of Technology Roorkee.

## References

[1] W. C. Lin, D. Y. Liao, C. Y. Liu, and Y. Y. Lee, “Daily imaging scheduling of an earth observation satellite,” *IEEE Trans. Syst., Man, Cybern. A, Syst. Humans*, vol. 35, no. 2, pp. 213–223, Mar. 2005.

[2] Y. Li, R. Wang, and M. Xu, “Scheduling and rescheduling of imaging satellite based on ant colony optimization,” *J. Comput. Inf. Syst.* vol. 9, no. 16, pp. 6503–6510, 2013.

[3] G. Langella, A. Basile, A. Bonfante, and F. Terribile, “High-resolution space–time rainfall analysis using integrated ANN inference systems,” *J. Hydrol.*, vol. 387, no. 3/4, pp. 328–342, Jun. 15, 2010.

[4] J. Li, X. Huang, and J. Gong, “Deep neural network for remote-sensing image interpretation: Status and perspectives,” *Nat. Sci. Rev.*, May 2, 2019.

[5] N. Valizadeh *et al.*, “Artificial intelligence and geo-statistical models for stream-flow forecasting in ungauged stations: State of the art,” *Natural Hazards*, vol. 86, pp. 1377–1392, 2017.

[6] X. Zhao, T. Xu, Y. Fu, E. Chen, and H. Guo, “Incorporating spatio-temporal smoothness for air quality inference,” in *Proc. IEEE Int. Conf. Data Mining*, 2017, pp. 1177–1182.

[7] Q. Liu, S. Wu, L. Wang, and T. Tan, “Predicting the next location: A recurrent model with spatial and temporal contexts,” in *Proc. 30th AAAI Conf. Artif. Intell.*, 2016, pp. 194–200.

[8] J. Ehrhardt, D. Säring, and H. Handels, “Structure-preserving interpolation of temporal and spatial image sequences using an optical flow-based method,” *Methods Inf. Med.*, vol. 46, no. 3, pp. 300–307, 2007.

[9] J. Ehrhardt, D. Säring, and H. Handels, “Interpolation of temporal image sequences by optical flow based registration,” in *Bildverarbeitung für die Medizin*, H. Handels, J. Ehrhardt, A. Horsch, H. P. Meinzer, and T. Tolxdorff, Eds. Berlin, Germany: Informatik aktuell Springer, 2006, pp. 256–260.

[10] X. Shi, Z. Chen, H. Wang, D.-Y. Yeung, W. K. Wong, and W.-C. Woo, “Convolutional LSTM network: A machine learning approach for precipitation nowcasting,” in *Advances in Neural Information Processing Systems*, vol. 28, C. Cortes, N. D. Lawrence, D. D. Lee, M. Sugiyama, and R. Garnett, Eds. Red Hook, NY, USA: Curran, 2015, pp. 802–810.

[11] A. Grover, A. Kapoor, and E. J. Horvitz, “A deep hybrid model for weather forecasting,” in *Proc. 21st ACM SIGKDD Int. Conf. Knowl. Discovery Data Mining*, Aug. 10–13, 2015, pp. 379–386.

[12] A. Kamilaris and F. X. Prenafeta-Boldú, “Deep learning in agriculture: A survey,” *Comput. Electron. Agriculture*, vol. 147, pp. 70–90, Apr. 2018.

[13] M. Hossain, B. Rekabdar, S. J. Louis, and S. Dascalu, “Forecasting the weather of Nevada: A deep learning approach,” in *Proc. Int. Joint Conf. Neural Netw.*, Killarney, Ireland, Jul. 12–17, 2015.

[14] J. Zhang, P. Zhong, Y. Chen, and S. Li, “$L_{(1/2)}$-regularized deconvolution network for the representation and restoration of optical remote sensing images,” *IEEE Trans. Geosci. Remote Sens.*, vol. 52, no. 5, pp. 2617–2627, May 2014.

[15] W. Huang, L. Xiao, Z. Wei, H. Liu, and S. Tang, “A new pan sharpening method with deep neural networks,” *IEEE Geosci. Remote Sens.*, vol. 12, no. 5, pp. 1037–1041, May 2015.

[16] X. Jia, B. C. Kuo, and M. M. Crawford, “Feature mining for hyperspectral image classification,” *Proc. IEEE*, vol. 101, no. 3, pp. 676–697, Mar. 2013.

[17] F. Melgani and L. Bruzzone, “Classification of hyperspectral remote sensing images with support vector machines,” *IEEE Trans. Geosci. Remote Sens.*, vol. 42, no. 8, pp. 1778–1790, Aug. 2004.

[18] L. Zhang, L. Zhang, and B. Du, “Deep learning for remote sensing data: A technical tutorial on the state of the art,” *IEEE Geosci. Remote Sens. Mag.*, vol. 4, no. 2, pp. 22–40, Jun. 2016.

[19] J. A. Parker, R. V. Kenyon, and D. E. Troxel, “Comparison of interpolating methods for image resampling,” *IEEE Trans. Med. Imag.*, vol. MI-2, no. 1, pp. 31–39, Mar. 1983.

[20] T. Gurdan, M. R. Oswald, D. Gurdan, and D. Cremers, “Spatial and temporal interpolation of multi-view image sequences,” in *German Conference on Pattern Recognition* (Lecture Notes in Computer Science), vol. 8753, X. Jiang, J. Hornegger, and R. Koch, Eds. Cham, Switzerland: Springer, 2014.

[21] J. L. Barron, D. J. Fleet, and S. S. Beauchemin, “Performance of optical flow techniques,” *Int. J. Comput. Vision*, vol. 12, pp. 43–77, 1994.

[22] B. K. P. Horn and B. G. Schunck, “Determining optical flow,” *Artif. Intell.*, vol. 17, no. 1–3, pp. 185–203, 1981.

[23] B. Lucas and T. Kanade, “An iterative image registration technique with an application to stereo vision,” in *Proc. 7th Int. Joint Conf. Artif. Intell.*, Vancouver, BC, Canada, 1981, pp. 674–679.

[24] L. M. G. Drummond and B. F. Svaiter, “A steepest descent method for vector optimization,” *J. Comput. Appl. Math.*, vol. 175, no. 2, pp. 395–414, 2005.

[25] J. Barzilai and J. M. Borwein, “Two-point step size gradient methods,” *IMA J. Numer. Anal.*, vol. 8, no. 1, pp. 141–148, 1988.

[26] R. Fletcher, *Practical Methods of Optimization*, 2nd ed. New York, NY, USA: Wiley, 1987.

[27] D. Marquardt, “An algorithm for least-squares estimation of nonlinear parameters,” *SIAM J. Appl. Math.*, vol. 11, no. 2, pp. 431–441, 1963.

[28] R. Fletcher and C. M. Reeves, “Function minimization by conjugate gradients,” *Comput. J.*, vol. 7, no. 2, pp. 149–154, 1964.

[29] W. C. Davidon, “Variable metric method for minimization,” *SIAM J. Optim.*, vol. 1, pp. 1–17, 1991.

[30] T. Brox, A. Bruhn, N. Papenberg, and J. Weickert, “High accuracy optical flow estimation based on a theory for warping,” in *Proc. Eur. Conf. Comput. Vision*, 2004, vol. 4, pp. 25–36.

[31] A. Bruhn, J. Weickert, and C. Schnorr, “Lucas/Kanade meets Horn/Schunck: Combining local and global optic flow methods,” *Int. J. Comput. Vision*, vol. 61, no. 3, pp. 211–231, 2005.

[32] P. R. Giaccone and G. A. Jones, “Feed-forward estimation of optical flow,” in *Proc. IET Conf.*, 1997, 204–208.

[33] J. Malo, J. Gutierrez, I. Epifanio, and F. J. Ferri, “Perceptually weighted optical flow for motion-based segmentation in MPEG-4 paradigm,” *Electron. Lett.*, vol. 36, no. 20, pp. 1693–1694, 2000.

[34] P. Kumar, S. Kumar, and R. Balasubramanian, "A fractional order variational model for the robust estimation of optical flow from image sequences," *Optik—Int. J. Light Electron Opt.*, vol. 127, no. 20, pp. 8710–8727, 2016.

[35] P. Kumar and S. Kumar, "A modified variational functional for estimating dense and discontinuity preserving optical flow in various spectrum," *AEU—Int. J. Electron. Commun.*, vol. 70, no. 3, pp. 289–300, 2016.

[36] M. Shi and V. Solo, "Empirical choice of smoothing parameters in robust optical flow estimation," in *Proc. IEEE Int. Conf. Acoust. , Speech, Signal Process.*, 2003, vol. 3, pp. 349–352.

[37] J. Redmon, S. Divvala, R. Girshick, and A. Farhadi, "You only look once: Unified, real-time object detection," in *Proc. IEEE Conf. Comput. Vision Pattern Recognit.*, 2016, pp. 779–788.

[38] H. H. Nagel and W. Enkelmann, "An investigation of smoothness constraints for the estimation of displacement vector fields from image sequences," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. PAMI-8, no. 5, pp. 565–593, Sep. 1986.

[39] S. Shakya and S. Kumar, "Characterising and predicting the movement of clouds using fractional-order optical flow," *IET Image Process.*, vol. 13, no. 8, pp. 1375–1381, 2019.

[40] W. R. Crum, O. Camara, and D. J. Hawkes, "Methods for inverting dense displacement fields: Evaluation in brain image registration," in *Proc. 10th Int. Conf. Med. Image Comput. Comput. Assisted Intervention—Part I*, 2007, pp. 900–907.

[41] D. C. Wilson and B. A. Mair, "Thin-plate spline interpolation," in *Sampling, Wavelets, and Tomography. Applied and Numerical Harmonic Analysis*, J. J. Benedetto and A. I. Zayed, Eds. Boston, MA, USA: Birkhäuser, 2004.

[42] T. Mühlenstädt and S. Kuhnt, "Kernel interpolation," *Comput. Statist. Data Anal.*, vol. 55, no. 11, pp. 2962–2974, 2011.

[43] S. Kumar, S. Kumar, N. Sukavanam, and R. Balasubramanian, "Dual tree fractional quaternion wavelet transform for disparity estimation," *ISA Trans.*, vol. 53, no. 2, pp. 547–559, 2014.

[44] [Online]. Available: https://keras.io/applications/

[45] [Online]. Available: https://colab.research.google.com/notebooks/welcome.ipynb#

[46] F. Chollet, "Xception: Deep learning with depthwise separable convolutions," in *Proc. Conf. Comput. Vision Pattern Recognit.*, 2017, pp. 1800–1807.

[47] J. D. Redmon, S. K. Divvala, R. B. Girshick, and A. Farhadi, "You only look once: Unified, real-time object detection," 2015, *arXiv:1506.02640*.

[48] S. Ren, K. He, R. B. Girshick, and J. Sun, "Faster R-CNN: Towards real-time object detection with region proposal networks," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 39, no. 6, pp. 1137–1149, Jun. 2017.

[49] T.-Y. Lin, P. Goyal, R. B. Girshick, K. He, and P. Dollár, "Focal loss for dense object detection," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 42, no. 2, pp. 318–327, Feb. 2020.

[50] [Online]. Available: http://satellite.imd.gov.in/archive/

[51] X. Yu X. Wu, C. Luo, and P. Ren, "Deep learning in remote sensing scene classification: A data augmentation enhanced convolutional neural network framework," *GISci. Remote Sens.*, vol. 54, no. 5, pp. 741–758, 2017.

[52] D. P. Huttenlocher, G. A. Klanderman, and W. J. Rucklidge, "Comparing images using the Hausdorff distance," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 15, no. 9, pp. 850–863, Sep. 1993.

[53] T.-L. Pao and J.-H. Yeh, "Typhoon locating and reconstruction from the infrared satellite cloud image," *J. Multimedia*, vol. 3, pp. 45–50, 2008.

[54] [Online]. Available: http://www.rsmcnewdelhi.imd.gov.in/images/pdf/publications/preliminary-report/helen_2013.pdf

[55] [Online]. Available: https://matplotlib.org/

[56] [Online]. Available: https://keras.io

[57] J. M. Johnson and T. M. Khoshgoftaar, "Survey on deep learning with class imbalance," *J. Big Data*, vol. 6, 2019, Art. no. 27.

[58] [Online]. Available: https://www.mosdac.gov.in/

[59] "JTWC track: Naval Meteorology and Oceanography Command" [Online]. Available: https://www.metoc.navy.mil/jtwc/jtwc.html?north-indian-ocean

[60] "National Center for Atmospheric Research database" [Online]. Available: http://hurricanes.ral.ucar.edu/realtime/plots/northindian/2017/io032017/

[61] M. Samy, S. K. Karthikeyan, S. Durai, and R. Sheriff, "Ockhi cyclone and its impact in the Kanyakumari district of Southern Tamilnadu, India: An aftermath Analysis," *Int. J. Recent Res. Aspects*, pp. 466–469, Apr. 2018.

[62] A. A. Fousiya and A. M. Lone, "Cyclone Ockhi and its impact over Minicoy Island, Lakshadweep, India," *Current Sci.*, vol. 115, no. 5, pp. 819–820, Sep. 10, 2018.

**Snehlata Shakya** received the M.Sc. degree from Nehru College Chhibramau, Chhatrapati Shahu Ji Maharaj University, Kanpur, India, in 2006, and the Ph.D. degree in mechanical engineering from the Indian Institute of Technology Kanpur, Kanpur, India, in 2014.

She has been working with the University of Bergen, Bergen, Norway and University of Linkoping, Linkoping, Sweden on various assignments. She is currently a Postdoctoral Employee with the Department of Clinical Physiology, Lund University, Lund, Sweden. Her research interests include computerized tomography, magnetic resonance imaging, medical imaging, image processing, inverse problems, error estimates, and fluid flow analysis.

**Sanjeev Kumar** received the M.Sc. degree in applied mathematics and Ph.D. degree in mathematics from the Indian Institute of Technology Roorkee, Roorkee, India, in 2003 and 2008, respectively.

He is currently an Associate Professor with the Department of Mathematics, Indian Institute of Technology Roorkee. His research interests include computer vision and mathematical imaging, inverse problems in imaging and control, and machine learning.

**Mayank Goswami** received the B.E. degree in electrical engineering from Madhav Institute of Technology and Science, Gwalior, India, in 2006, and the M.Tech. and Ph.D. degrees in nuclear engineering and technology from the Indian Institute of Technology Kanpur, Kanpur, India, in 2009 and 2014, respectively.

He is currently an Assistant Professor with the Department of Physics, Indian Institute of Technology Roorkee, Roorkee, India. His research interests include image and signal processing, instrumentation for noninvasive and direct imaging, and algorithm development for inverse problem using AI and classical optimization.