The Astronomical Journal, 159:212 (10pp), 2020 May https://doi.org/10.3847/1538-3881/ab800a

Detection and Classification of Astronomical Targets with Deep Neural Networks in Wide-field Small Aperture Telescopes

Peng Jia[1,2], Qiang Liu[1], and Yongyang Sun[1]
[1] College of Physics and Optoelectronics, Taiyuan University of Technology, Taiyuan, 030024, People's Republic of China; robinmartin20@gmail.com
[2] Key Laboratory of Advanced Transducers and Intelligent Control Systems, Ministry of Education and Shanxi Province, Taiyuan University of Technology, Taiyuan, 030024, People's Republic of China
Received 2020 January 2; revised 2020 March 12; accepted 2020 March 14; published 2020 April 16

Abstract

Wide-field small aperture telescopes are widely used for optical transient observations. Detection and classification of astronomical targets in observed images are the most important and basic step. In this paper, we propose an astronomical target detection and classification framework based on deep neural networks. Our framework adopts the concept of the Faster R-CNN and uses a modified Resnet-50 as a backbone network and a feature pyramid network to extract features from images of different astronomical targets. To increase the generalization ability of our framework, we use both simulated and real observation images to train the neural network. After training, the neural network could detect and classify astronomical targets automatically. We test the performance of our framework with simulated data and find that our framework has almost the same detection ability as that of the traditional method for bright and isolated sources and our framework has two times better detection ability for dim targets, albeit all celestial objects detected by the traditional method can be classified correctly. We also use our framework to process real observation data and find that our framework can improve 25% detection ability than that of the traditional method when the threshold of our framework is 0.6. Rapid discovery of transient targets is quite important and we further propose to install our framework in embedded devices such as the Nvidia Jetson Xavier to achieve real-time astronomical targets detection and classification abilities.

Unified Astronomy Thesaurus concepts: Classification (1907); Convolutional neural networks (1938); Transient detection (1957); Astrostatistics techniques (1886)

1. Introduction

In recent years, observing astronomical targets with variable magnitude or position has become an active research area. These targets, usually known as transients, require observations with adequate spatial and temporal resolution. To satisfy these requirements, wide-field small aperture telescopes (WFSATs) are commonly used (Burd et al. 2005; Ping & Zhang 2017; Ratzloff & Law 2019; Xu et al. 2020). Because WFSATs have small aperture and a wide field of view, we can use several of them to build an observation net in a cost effective way. In an observation net, WFSATs are normally placed in different sites or in the same site pointing to different directions. The observation net can provide acceptable sky and temporal coverage with moderate cost. Because all these WFSATs are observing the sky in sidereal tracking mode with relatively short exposure times, there are huge amount of data to be processed each night. It is a time intensive task to process these data, since some transients require immediate follow-up observations through spectroscopy or imaging in other bands or by larger telescopes.

For transient observation tasks, data processing in WFSATs normally include three steps: extracting images of candidate astronomical targets with a source extracting method (detection), classifying these candidates into different categories (classification), and cross-matching these targets with catalogs, such as the Tycho 2 Catalog for celestial objects (Høg et al. 2000). The efficiency and the effectiveness of the first two steps usually limit the observation ability of WFSATs, because we cannot report effective information about a transient candidate if we cannot detect or classify it correctly. The state-of-the-art detection method is scanning the whole frame of observation images with a source extracting algorithm. Based on our experience, the most robust detection algorithm is that in SExtractor proposed by Bertin & Arnouts (1996).

After detection, a lot of candidate astronomical images can be extracted and classification algorithms are required to classify these candidates. In recent years, different machine-learning-based astronomical image classification algorithms have been developed and they have achieved higher and higher classification accuracy and recall rates (Romano et al. 2006; Gonzalez et al. 2018; Tachibana & Miller 2018; Burke et al. 2019; Duev et al. 2019a, 2019b; Mahabal et al. 2019; Turpin et al. 2020). For WFSATs, we have also proposed a transient classification method based on ensemble learning and neural networks (Jia et al. 2019). Our method can achieve acceptable classification performance for different kinds of astronomical targets. As more and more new image classification algorithms are proposed, we could expect better classification algorithms in the future.

However, all astronomical targets should be detected before they can be classified. Simply increasing the performance of the classification algorithm is not enough to increase the scientific output of WFSATs. Assuming even if the classification algorithm could obtain ideal classification accuracy and recall rate, targets that cannot be detected by the detection algorithm would never be processed, which will limit the observation ability of WFSATs. For example, the classification method proposed in Jia et al. (2019) can achieve a human-level classification ability for different kinds of celestial objects detected by the classical SExtractor-based algorithms. However, some astronomical targets such as moving targets, which have streak-like images, or point-like astronomical images with a low signal-to-noise ratio (S/N) cannot be detected and they

will never be processed as celestial objects. If there are transients in these celestial objects, they will never be observed. Meanwhile although these targets cannot be detected, our classification neural network can achieve almost the same accuracy and recall rate for these targets as that for other astronomical targets.

This problem is probably caused by the trade-off between versatility and specificity. For general purpose source detection algorithms, we set general parameters. Then these algorithms have a robust detection ability for different kinds of astronomical targets at the expense of a low detection rate for a special kind of astronomical targets. For example, the detection algorithm in SExtractor uses a very elegant rule: a group of connected pixels with values exceeding a predefined value will be recognized as a detection. It can give promising results for different types of sources, but for extended targets with low S/N or multiple blended targets, the detection performance will drop down. However neural-network-based astronomical image classification algorithms classify images according to their overall structure. They can achieve relatively higher classification accuracy and recall rate for candidates with low S/N or several candidates that are close to each other. So if we integrate the classification algorithms with the detection algorithms, we can design a framework which would increase the performance of WFSATs in transient detection tasks.

There are already several state-of-the-art detection and classification frameworks, such as the Faster R-CNN (Ren et al. 2017), You Only Look Once (YOLO; Redmon et al. 2016), and the Single Shot MultiBox Detector (SSD; Liu et al. 2016). For astronomical targets, Gonzalez et al. (2018) propose DARKNET, which uses YOLO for real-time galaxy detection and identification. Duev et al. (2019a) propose to use the machine-learning algorithm to classify star–galaxy and separate real/bogus transients. Burke et al. (2019) use a semantic segmentation model named Mask R-CNN (He et al. 2017) for real-time astronomical targets detection and classification. These methods mentioned above are mainly used for general purpose sky survey telescopes and they require a large amount of astronomical images that are labeled by human experts as training data. Obtaining training data is hard for general purpose survey telescopes, because it require a large amount of human labeling. For WFSATs, it would be harder, because there are a lot of WFSATs and only a limited number of images obtained by them will be checked by scientists, albeit labeling these images. Besides, images obtained by WFSATs have a low S/N and low-spatial sampling rate. A new detection and classification framework is required.

In this paper, we propose to adopt the concept of Faster R-CNN to build an astronomical target detection and classification framework for WFSATs. Our framework uses a feature pyramid network architecture to extract features from images and a modified Resnet-50 as a backbone network for classification and regression. We use simulation data to test the performance of our framework and find that our framework is better in the detection of extended targets or astronomical targets with low S/N. For real applications, we propose to use simulated images and real observation images to train our framework. After training, we find that our framework has a better performance than the classic framework. To further increase transit detection ability of WFSATs, we further propose to install our framework in an embedded device to achieve real-time observation ability.

This paper is organized as follows. In Section 2, we introduce the transient observation task and the classic detection and classification framework in WFSATs. In Section 3, we introduce our approach and we compare the performance of our framework and that of the classic framework in Section 4. In Section 5, we show how to apply our framework to real observation data obtained by WFSATs through transfer learning, which means we use new observation data to train our neural network that are already trained with simulated images (Zhuang et al. 2019). We make our conclusions and propose our future work in Section 5.

2. Astronomical Target Detection and Classification Task and Classic Methods in WFSATs

In WFSATs, transient observation tasks include discovery and observation of astronomical targets with temporal-varying properties, such as astronomical targets with magnitude or position variation. Because WFSATs are low-cost, we can use several of them to build an observation net to achieve high sky and temporal coverage. To reduce the total cost and maximize detection ability, our WFSATs are working in white-light mode (no filter) and there is no field de-rotator installed in them. Due to this design concept, images obtained by our WFSATs are affected by color aberration and field rotation (Jia et al. 2020a), which make traditional image-difference-based data processing methods hard to apply (Zackay et al. 2016). So in real applications, we directly process the original observation images with the following steps.

1. We select effective images that are not seriously affected by clouds or the malfunction of cameras with the support of a vector machine algorithm (Li-wen et al. 2019).
2. We scan the whole frame of the selected images with a source detection method to locate candidate astronomical targets. The source detection algorithm in SExtractor is commonly used and we set very low values of the threshold (1.1σ) and connected area (3 pixels) to make sure that all targets can be detected.
3. All the images of candidate astronomical targets are sent to an astronomical image classification algorithm and we will classify these targets into different categories. After this step, we could obtain the positions and types of all the candidate astronomical targets.
4. We cross-match different candidate astronomical targets with different catalogs. Because the diameter of our telescope is small and the exposure time is short, we will only cross-match candidate astronomical targets with bright source catalogs. After this step, all the unmatched images will be identified as transient candidates and they will be reported to data center for next step observations.

With the data processing framework mentioned above, WFSATs are capable of observing bright transits, such as nearby supernova, tidal disruption flares, comets, meteors, asteroids, or space debris. Although the optical design of WFSATs can guarantee good image quality, normally atmospheric turbulence becomes the major aberration contribution (Jia et al. 2020b); it is very hard to find science cameras with low noise, high speed, and a large number of pixels at the same time, which would lead to low-spatial sampling rates in the image plane of WFSATs. Images of astronomical targets with a

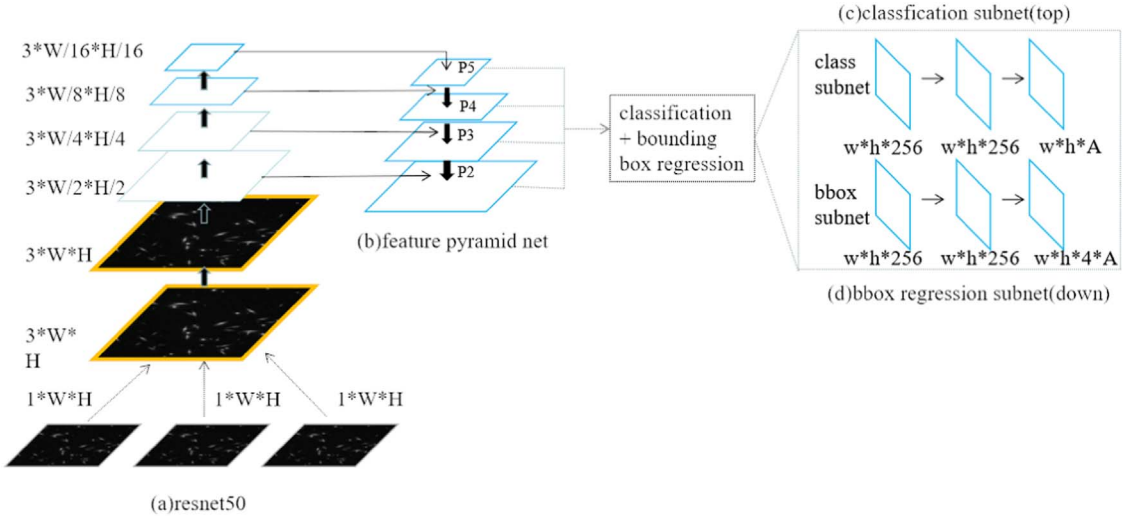

Figure 1. Architecture of the Faster R-CNN used in this paper. An input image includes three channels and the image in each channel is the same astronomical image. The input image passes into the first convolutional layer, which has three convolution layers with a stride of one and and a convolution kernel of 3×3 to process each channel of the input image. Then the output of the first convolutional layer will be put into the feature pyramid network for feature extraction. In the feature pyramid network, feature maps from the P2 layer are used as features of the images for each candidate. These feature maps will be used for classification by comparison between the features of different categories and they will also be used for position regression through bounding box regression. The Rectified Linear Unit (ReLU) function, which uses the rectifier activation function $\max(0, n)$ to evaluate outputs of previous layers, is used as activation functions for all five hidden layers. In this figure, W and H stand for the size of original image and w and h stand for the size of the candidate image. The blue boxes stand for the different convolutional stages and the shape of these layers is shown beside these boxes.

low-spatial sampling rate are easily mistaken as background variation or cosmic rays.

To reduce the effects brought on by the aforementioned problems, we develop a framework that includes the detection algorithm in SExtractor and the neural-network-based astronomical target classification algorithm (Jia et al. 2019). It can achieve more than 94% classification accuracy for different types of astronomical targets, but we find a problem in this framework that is caused by a mismatching between the classification algorithm and the detection algorithm. The recall rate of the detection method is low for extended targets and targets with low S/N. It means that many astronomical targets, which can be classified correctly, have been ruled out in the detection step. This problem is vital to WFSATs because it would reduce the detection ability of the whole observation net. Considering that the machine-learning-based classification algorithm has already proven its good performance in classifying between different targets, we propose to develop new framework based on the classification algorithm, which can detect and classify images of different astronomical targets at the same time. In the next section, we will introduce our framework.

3. Astronomical Target Detection and Classification Framework based on Deep Neural Networks

Because the spatial sampling rate in WFSATs is low (several arcseconds per pixel) and the exposure time is short (around several seconds), images obtained by WFSATs are quite different from images obtained by general purpose sky survey telescopes. For ordinary stars with moderate S/N, images normally have around 5×5 pixels. For bright stars or fast moving targets, the images will extend to tens of pixels. This would lead to a different classification and detection framework compared to the framework proposed for general purpose sky surveys.

3.1. Architecture of the Astronomical Target Detection and Classification Framework for WFSATs

The framework proposed in this paper adopts the concept of the Faster R-CNN neural network (Ren et al. 2017). It uses several windows with a predefined size to scan the whole frame of an image and small images obtained by these windows will be sent into neural networks for classification and regression. The classification results will indicate different types of astronomical targets and the regression results will provide positions of these astronomical targets. The overall architecture of our framework is shown in Figure 1 and it can be divided into four parts as shown in Figure 3.

1. Feature Extraction. In this part, neural networks are used to extract features of images for a subsequent region proposal network layer (RPN), which will propose candidate regions that contain celestial objects for classification and position regression, and a full connection layer (FC). Convolutional neural networks are commonly used to extract features from images. In this paper, we choose the feature pyramid network (FPN) structure, which consists of convolutional layers, ReLU activation layers, and pooling layers, as shown in left part of Figure 1. Because the size of the astronomical images obtained by WFSATs is small, if we use CNN with multiple layers for feature extraction, features and spatial location will be dispersed quickly. Besides because images of candidate astronomical targets have different sizes than that of the perceptive filed, the classification accuracy will drop down (Zhang et al. 2019a). To solve these problems, we use a customized Resnet-50 as the backbone of our neural network. We also use a convolutional kernel with a size of 3×3 to replace the original convolutional kernel with a size of 7×7 in the Resnet-50, as shown in Figure 2.

2. Region Proposal. The region proposal stands for a proposal of small regions that contain celestial objects for subsequent networks. In this part, we will first divide the original image into small images of different size and shape

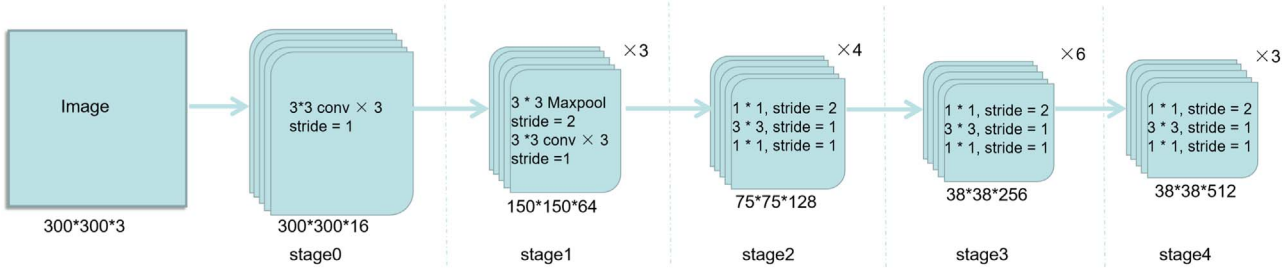

Figure 2. Structure of the Resnet-50 used in this paper. It includes five convolutional stages shown as boxes and in each stage there are several convolutional layers. The size of the convolutional kernel in these convolutional layers is shown as $n \times n$. The size of each stage is shown below each box with width × length × channel.

with predefined values. These small images are called anchors. Then feature maps of anchors are extracted by the FPN as we discussed in part 1. The soft-max classifier, which is a function that normalizes input vectors into a probability distribution with probabilities proportional to the exponential of the inputs (Bishop & Nasrabadi 2007), will classify these anchors into either the background or the target. Then the RPN will change position and the size of anchors, labeled as targets continuously through bounding box regression and output shifted and rescaled anchors, which have maximal possibility to be astronomical targets as proposals. Normally in this step a rough estimation of the coordinates of these candidate targets could be obtained.

3. ROI Pooling/Alignment. With region proposals and their corresponding feature maps, the region of interest (ROI) pooling/alignment part maps proposals with different sizes into the same size. Then feature maps of rescaled proposals will be sent into the FC for classification. The ROI pooling, which transfers the maximum values within some predefined sections to the next level, is widely used in general detection frameworks (Girshick et al. 2013), such as Fast R-CNN, Faster R-CNN, or RFCN. The ROI pooling will pool the corresponding areas into feature maps of fixed size according to predefined selection boxes. Since the size of feature maps after pooling must be a constant, there are two quantization procedures for ROI Pooling.

1. When images are passed into the backbone network, the boundary of the candidate box will be quantized to the same size of the feature maps. Information in the boundary parts will be lost during this process.
2. During the pooling process, input images will be divided into units with a size of $k \times k$ pixels and information in the boundary of each unit is lost.

These two quantization processes will introduce a loss of regression accuracy in the candidate regions, especially for small targets. For example, an image with a 0.1 pixel shift in the backbone network would introduce a 1.6 pixel shift in the original image, if we quantize images to $1/16$ of the original images in the backbone network. Considering images of astronomical targets obtained by WFSATs normally only have 5×5 pixels, the information loss would become a serious problem. The ROI alignment is proposed to solve this problem (He et al. 2017), which includes the following modifications.

1. We will iterate over each region proposal to keep floating point boundaries accurate.
2. The region proposal is divided into $k \times k$ units and we do not quantize the boundaries of each unit.

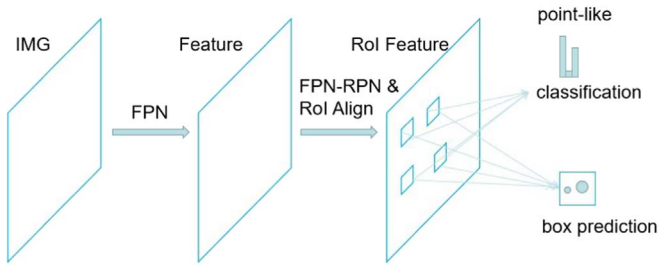

Figure 3. Flow chart of neural network used for the astronomical target detection and classification framework. The framework uses the FPN to extract features from the original image. Then with the RPN and ROI alignment, feature maps of candidate images are transmitted to the classification and regression neural network. Through box regression and classification, the neural network will classify and obtain the position of these targets. In this figure, boxes stand for the manipulation and arrows stand for the data flow.

3. Four fixed coordinate positions are calculated in each cell and values of these four positions are calculated through a bilinear interpolation method, which evaluates values according to the distance between known coordinate positions and their values (Seiler & Seiler 1989). Then we will carry out the maximum pooling operation.

The first and the second modification keep the regression results accurate enough, while the third modification keeps the pooling results of the backbone network accurate. In our framework, we use the ROI alignment instead of the ROI pooling.

4. Classification and Regression. In this step, feature maps of proposals are sent into the classification neural network for classification. Meanwhile, proposals will be rescaled and shifted again through bounding box regression to achieve higher regression and classification accuracy. At last, we will output the final positions and types of these proposals. After this step, the position and type of all candidate astronomical images are obtained by our framework.

3.2. Implementation of Our Astronomical Target Detection and Classification Framework with Simulated Data

3.2.1. Data Preparing

To test the performance of our framework in the detection and classification of astronomical targets, we generate mock observation data of WFSATs through Monte Carlo simulation. Because our WFSATs are working in sidereal tracking mode, there are two different astronomical targets in the observation data: point-like sources and streak-like sources. Point-like sources are normally stars, supernovae, or tidal disruption flares. Streak-like sources are moving objects, such as comets,

Table 1
Magnitude and Ground Truth Box Size in Pixels

Type	Magnitude	Size
Point-like	Brighter than 8	24 × 24
	9–10	12 × 12
	10–23	10 × 10
Streak-like	19–23	12 × 12

meteors, asteroids, or space debris. The length of a streak-like source is defined by its moving speed and exposure time and we assume that the streak-like source has a moving speed of around 15″–50″ s^{-1} in this paper.

We use the SkyMaker (Bertin 2009) to generate mock observation data. First of all, we assume that the astronomical objects are static in a very short time period (around 5 ms) and we generate positions and magnitudes of these targets in every instantaneous moment. Then we use SkyMaker to generate images in these moments and, according to exposure time, we stack these images to generate mock observation data. In our simulation, the telescope has a field of view of 10 arcmin and a pixel scale of 1″ per pixel. The magnitude of the sky background is 25 and the FWHM of the seeing disk is 1.″0. The exposure time is 1 s for each frame with a readout noise of 1 e^- and a dark current of 1 e^- s^{-1}.

3.2.2. Data Labeling

Labeled images, which include the original images and the labels indicate the positions and types of targets, are required to train our framework. In this paper, positions of targets are labeled in the original images by a small rectangular box called the ground truth box. For an astronomical target i, its ground truth box contains information stored in a vector with four dimensions: $X_{\rm ih}$, $X_{\rm il}$, $Y_{\rm ih}$, and $Y_{\rm il}$. $X_{\rm ih}$ and $X_{\rm il}$ stand for maximal and minimal values of coordinates of the target image along the X direction, while $Y_{\rm ih}$ and $Y_{\rm il}$ stand for maximal and minimal values of coordinates of the target image along the Y direction. Normally training data are labeled manually. However, since we use SkyMaker to generate mock observation data in this paper, we already have the positions, magnitudes, and types of these astronomical targets in every simulated image. According to the information of these targets, we create ground truth boxes of every object with Equation (1). Because astronomical targets with different magnitude have a different size, σ is defined according to Table 1. An image of an astronomical target along with its ground truth box is shown in Figure 4.

$$\begin{aligned} X_{\rm ih} &= X_{\rm i} + \sigma \\ X_{\rm il} &= X_{\rm i} - \sigma \\ Y_{\rm ih} &= Y_{\rm i} + \sigma \\ Y_{\rm il} &= Y_{\rm i} - \sigma. \end{aligned} \tag{1}$$

3.3. Training of Our Framework

We use the following tactics to train our framework.

1. Data augmentation: for real observations, data labeling is quite expensive. Before sending images into our framework, we randomly rotate them to generate several new images to increase the number of training images and generalization ability of our framework. Besides, we normalize every frame of images with Equation (2):

$$\mathrm{Img}(i, j) = \frac{P(x_i, y_i) - \overline{P(x, y)}}{P(x, y)_{\max} - P(x, y)_{\min}} \tag{2}$$

where $P(x_i, y_i)$ is the gray-scale value in an image, $\overline{P(x, y)}$ is the average gray-scale value of an image, $P(x, y)_{\max}$ and $P(x, y)_{\min}$ are the maximum and minimum gray-scale values of an image, respectively, and $\mathrm{Img}(i, j)$ is the image used for neural network training.
2. We upsample all of the original images to two times their original size to increase the detection ability of our framework.
3. We use instance normalization, which subtracts the mean value and divides the variance of each image for each pixel in the image, to increase convergence speed during training (Ulyanov et al. 2016).
4. To prevent overfitting, we use L1 and L2 loss as regularization loss functions and we set the weight of the L2 loss as 0.00001 as shown in Equation (3),

$$\mathrm{Loss}(x, y) = \frac{1}{n}\sum_i \|x_i - y_i\| + 0.00001 * (x_i - y_i)^2. \tag{3}$$

5. We choose the CrossEntropy function as shown in Equation (4) as the loss function for the classification and smooth L1 loss function defined in Equation (5) for the bounding box regression,

$$L_H(p_i, y_i) = -\sum_{k=1}^{d} [p_{ik} \log_{y_{ik}} + (1 - p_{ik}) \log_{(1-y_{ik})}], \tag{4}$$

where p_{ik} stands for probability for celestial images of type y_i. The $L1_{\rm loss}(x, y)$ stands for the smoothed L1 loss function,

$$L1_{\rm loss}(x, y) = \frac{1}{n}\sum_i z_i, \tag{5}$$

where

$$z_i = \begin{cases} 0.5(x_i - y_i)^2, \text{ if} \|x_i - y_i\| < 1 \\ \|x_i - y_i\| - 0.5, \text{ otherwise.} \end{cases} \tag{6}$$

6. An anchor box is used to obtain small candidate images for further classification. The size of the anchor box

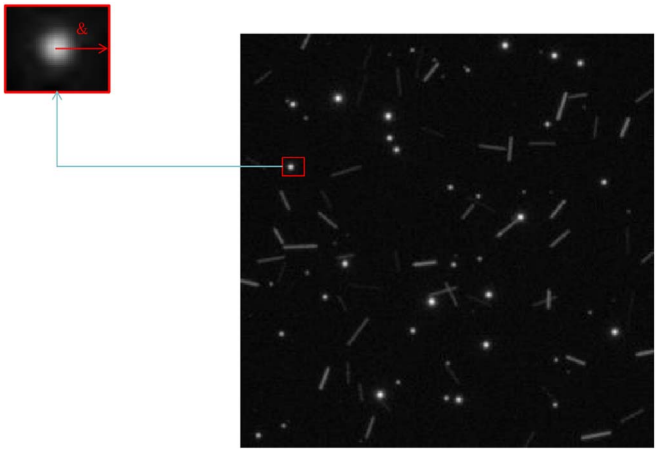

Figure 4. Ground truth box for an astronomical target. The ground truth box is created according to its center coordinate and σ. σ is different for astronomical targets with different magnitudes.

should be defined manually and match the size of the candidate images. Because there are different types of targets for detection, the size and the aspect ratio of the anchor box should be different. We use an anchor box with a size of [2, 4, 6] and an aspect ratio of [0.5, 1, 2] in this paper. Features of images will be extracted by the FPN and the size of the anchor box in different layers of the FPN is defined as [1, 2, 4, 8, 16].

The whole framework is implemented with Pytorch (Ketkar 2017) in a computer with two Nvidia GTX1080Ti graphics processor units (GPU). Of simulated data, 2000 frames are used as the training set and 500 frames of simulated data are used as the test set. We initialize our neural network with random weights and train our neural network for 20–30 epochs with the Adam algorithm (Kingma & Ba 2015) as the optimization algorithm. We set the learning rate with the warming-up method and the initial learning rate is set to 0.00003 (Zhang et al. 2019b).

4. Performance of Our Framework

4.1. Performance Evaluation Method

In this part, we will compare the performance of our framework with the classic detection and classification framework based on SExtractor. Because SExtractor is a source detection algorithm, we assume that all the targets detected by SExtractor can be correctly classified, which is an optimistic estimation. Furthermore, we set the detection and classification results as true positives when the detection result is correct and the location of the detected targets is within 1.5 pixels of the ground truth position. Otherwise, we set the detection as a false positive or a false negative. With this definition, we can directly compare the performance of different detection and classification frameworks. We use the mean average precision (mAP) to evaluate the performance of our framework and that of the classic method. Because of the real observed images, it is impossible to label all of the astronomical targets and there are only limited targets that can be labeled by human experts, it would be more practical to use precision to evaluate the performance of our framework. The mAP is defined in Equation (7) as average accuracy of all categories,

$$\mathrm{AP}_{11\mathrm{points}} = \frac{1}{11}\sum_{r=0}^{1.0} P_{\mathrm{interp}}(r)$$
$$\mathrm{mAP} = \frac{\sum_{q=1}^{Q} AP(q)}{Q}, \tag{7}$$

where $\mathrm{AP}_{11\mathrm{points}}$ represent the sum of the maximum precision $P_{\mathrm{interp}}(r)$ for 11 different recall values (e.g.,: $r = 0$, 0.1, 0.2, 0.3, 0.4, 0.5, 0.6, 0.7, 0.8, 0.9, 1.0) when the recall value is greater than a predefined threshold. Q is the number of classes of different astronomical targets. mAP is the mean value of all AP values corresponding to all categories. Because we know the position and type of all the astronomical targets, we can also use the recall rate and precision rate to evaluate the detection and classification results. The recall rate and precision rate are defined in Equation (8) as

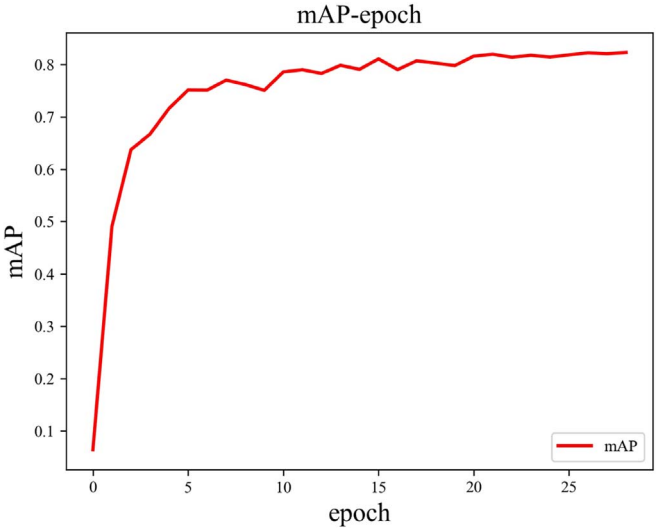

Figure 5. Mean average precision curve under the condition of the warming-up learning rate and a test set of 500 images, We have drawn the test precision of each epoch under 30 epochs. As we can see, the value of mAP began to stabilize around 0.8 after the 30th epoch.

$$\mathrm{recall} = \frac{\mathrm{TP}}{\mathrm{TP} + \mathrm{FN}}$$
$$\mathrm{precision} = \frac{\mathrm{TP}}{\mathrm{TP} + \mathrm{FP}} \tag{8}$$

where TP is the true positives (number of correctly classified positives), TN is the true negatives (number of correctly classified negatives), FP is the false positives (number of incorrectly classified positives), and FN is the false negatives (number of incorrectly classified negatives).

We will first show the performance of our framework. Figure 5 shows the curve of mAP values in the test set versus epoch number and we find that after 30 epochs, the mAP is stable. We use 30 epochs to train our framework in real applications. After training, the precision rate versus the recall rate curve is shown in Figure 6. We find that the accuracy is almost the same for two different targets, while the recall rate is slightly different. Our framework is better in detecting and classifying streak-like targets.

4.2. Comparison the Performance with Simulated Data

In this part, we will compare the performance of our framework with that of the classic framework. We use 200 simulated images as a validation set to test these two frameworks. Because we try to compare the performance of different detection and classification frameworks for targets with different S/N, we use a slightly different definition of the $\mathrm{precision}_{\mathrm{rate}}$ and the $\mathrm{recall}_{\mathrm{rate}}$ in this paper, as shown in Equation (9),

$$\mathrm{precision}_{\mathrm{rate}} = \frac{\mathrm{object}_{\mathrm{match}}}{\mathrm{object}_{\mathrm{total}}}\frac{\mathrm{object}_{\mathrm{total}}}{\mathrm{object}_{\mathrm{interval}}}$$
$$\mathrm{recall}_{\mathrm{rate}} = \frac{\mathrm{object}_{\mathrm{match}}}{\mathrm{object}_{\mathrm{Detect}}}, \tag{9}$$

where $\mathrm{object}_{\mathrm{match}}$ is the number of the detected targets that are correctly detected, $\mathrm{object}_{\mathrm{total}}$ is the total number of objects,

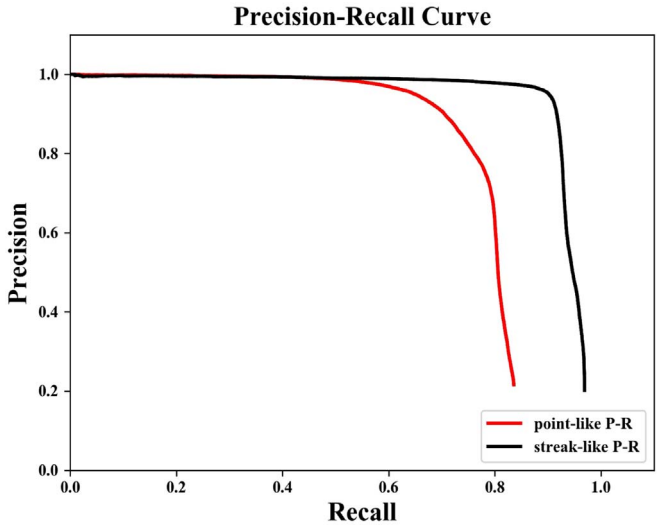

Figure 6. Precision–recall curve for the each transient category. The red line represents the precision–recall curve for point-like sources, while the black curve is the precision–recall curve for the streak-like sources. From this figure, we find that the Faster R-CNN has a different performance for different astronomical sources. The Faster R-CNN is better for the detection and classification of streak-like sources.

$\text{object}_{\text{interval}}$ is the number of objects within a certain range of magnitude, and $\text{object}_{\text{Detect}}$ is the number of all the detected targets. This definition reduces problems brought on by the unbalanced data set. We further use the $\text{precision}_{\text{rate}}$ and the $\text{recall}_{\text{rate}}$ to define the $f1$_score and the $f2$_score to compare the performance of these two methods, as shown in Equation (10). The $f1$_score is the harmonic mean of the precision and recall, while the $f2$_score is the weighted average of precision and recall. The $f1$_score is commonly used to evaluate the overall performance of the classification algorithm. The $f2$_score is a more conservative metric, because it assumes that the classification accuracy is more important than recall rate. In this paper, we use both of these two metrics to evaluate the performance of these two methods,

$$f1\_\text{score} = \frac{2 * \text{precision} * \text{recall}}{\text{precision} + \text{recall}}$$
$$f2\_\text{score} = \frac{(1 + 2 * 2) * \text{precision} * \text{recall}}{2 * 2 * \text{precision} + \text{recall}}. \quad (10)$$

One frame of astronomical images in the validation set and the detection results are shown in Figure 7. We find that these two methods can both detect astronomical images, but the performance of these two methods is obviously different. For astronomical targets near the bright target, our framework could still detect them, while the classic method is more likely to identify all these targets as one target. Meanwhile there are still several astronomical images that cannot be detected by our framework. This problem requires some new methods to further increase its detection ability and we will discuss that in our next paper.

Quantitative comparison of the performance of the classic framework and that of our framework is shown in Figure 8. We find that the recall rate for the bright sources are almost the same for both of these two frameworks. However for sources with low S/N, our framework has a much higher recall rate. For the accuracy rate, the classic framework is slightly better than that of our framework for bright sources, under the assumption that all sources detected by SExtractor can be correctly classified, which is almost impossible. When the S/N is low, the classic framework has lower classification accuracy. Overall, our framework performs better than the classic framework, as shown in Figure 9.

4.3. Comparison of the Performances with Real Observation Data

In this part, we use real observation data from one of our WFSATs to test these two frameworks. This WFSAT is a refractive telescope (Sun et al. 2019) with parameters shown in Table 2. To collect enough data for training and performance evaluation, we have set up a data annotation platform based on HTML and Java. Undergraduates from six classes are working together to annotate these images and we have collected over 600 frames of labeled images. In these images, all celestial objects are labeled and cross-checked with the catalogs. Then we train our framework with the following steps.

1. Select 600 real images that contain streak-like and point-like objects.
2. Select 75% of these images as training images and the rest of them as test images.
3. Estimate observation conditions and generate 500 frames of simulated images, which also contain streak-like and point-like objects.
4. The simulated images and the training images are regarded as the training set, while the test images are regarded as the test set.
5. Train our framework with the simulated images for 30 epochs as pre-training framework.
6. Train the pre-training framework with real observation images for 10 epochs.

After training, our framework can detect and classify astronomical targets automatically. We use the test set to test our framework and the classic framework. One frame of test images is shown in Figure 10. We find that our framework can detect and classify almost all astronomical targets robustly. Cosmic rays, hot pixels, and linear interference do not have any effect on the detection and classification results. We further compare the performance of our framework and that of the classic framework with 128 frames of validation images (images obtained by the same telescope, but on different days). Although astronomical targets in these images are checked by human experts and cross-matched by catalogs, it is still possible that there are targets that are not labeled, so we only show the improvement ratio of these two frameworks under different thresholds in Table 3. For the lower threshold, our framework can give results with a higher precision ratio and for the higher threshold, our framework can give results with a higher recall ratio. We find that our framework can achieve around a 1.032 improvement ratio, even when the threshold is as low as 0.4. If the threshold is 0.6, our framework can detect 25% more targets than that detected by the classic framework.

4.4. Applying our Framework in Embedded Devices

To further test the performance of our framework, we use the same star catalogs to generate simulated images with different

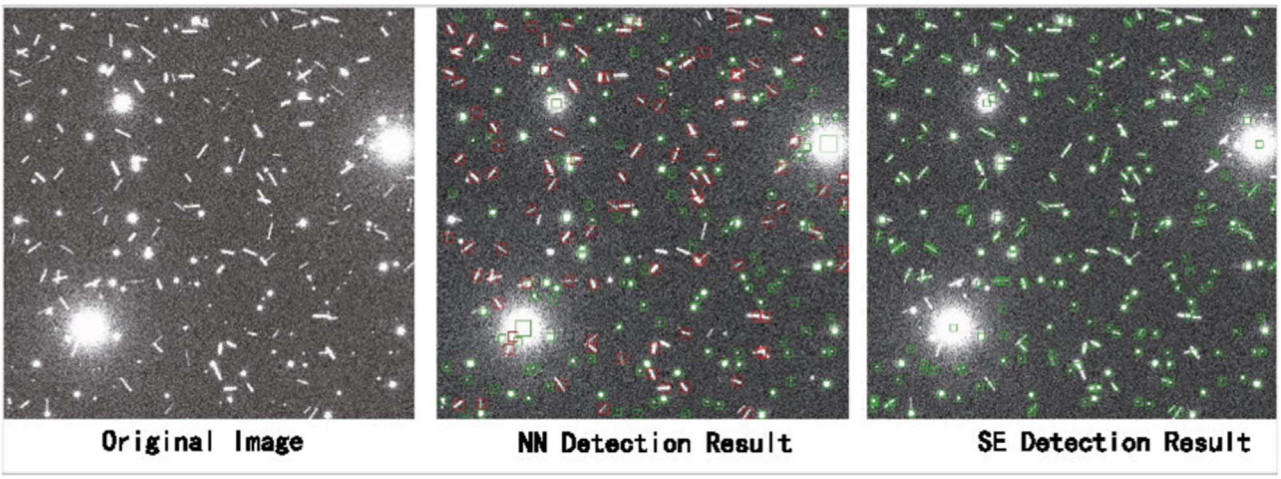

Figure 7. The first image is the original image which includes both point-like sources and streak-like sources. We transform the gray values in this figure to their log values for better visualization. The second image is the result obtained by our framework. Red boxes represent the streak-like astronomical images and green boxes represent point-like astronomical images. The third image is the result of detection by the classic framework. Because the classic framework cannot classify targets into different types, we assume all the astronomical targets detected by the classic framework can be correctly classified. As we can see in the third image, the classic framework does not perform well for blended sources, in addition, the streak-like target will be detected multiple times by the the classic framework. Therefore, when comparing the detection performance of our framework and that of the classic framework, targets that are detected multiple times are only calculated once.

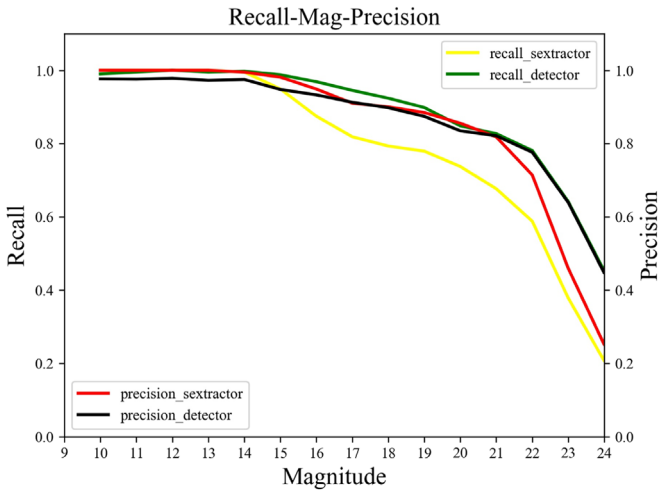

Figure 8. This figure shows how the recall rate and the precision rate will change for astronomical images with different S/N. The number of astronomical sources in each magnitude is around 2000–3000. From this figure, we find that the classic framework (labeled with SExtractor) has a slightly better performance for astronomical sources with higher S/N, while for astronomical sources with low S/N, our framework (labeled with the detector) is better.

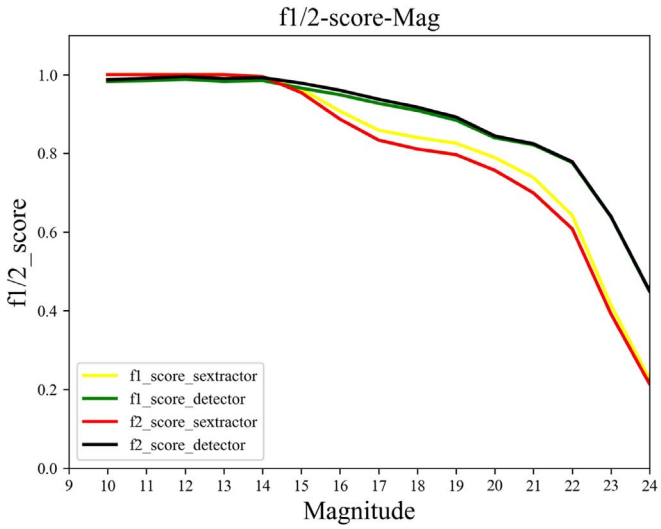

Figure 9. This figure shows the performance of these two frameworks for astronomical targets with different S/N, evaluated with the *f*1_score and the *f*2_score. We find that the performance of our framework (labeled with the detector) is better than that of the classic framework (labeled with SExtractor) for astronomical targets with low S/N and their performance is almost the same for bright sources.

random numbers. Since these images are generated with the star catalog and the same level of S/N, we could expect the same detection and classification results. However we find that detection and classification results will change, even though all these images contain almost the same astronomical information. If we use all these images for object detection and classification, we can get higher accuracy rate and recall rate. This property indicates that the best method to further increase the observation ability of WFSATs is to observe transient candidates several times as soon as we detect them. Because there are several WFSATs in an observation net, if we can achieve real-time detection and classification ability, the observation ability of the whole observation net can be further increased.

Based on this requirement, we propose to install our framework in an embedded device for each WFSAT to achieve real-time detection and classification ability. Since our framework uses deep neural networks, tensor cores are required to increase detection and classification speed. We install our

Table 2
Details of the Telescope

Parameter	Value
Aperture	500 mm
Field of view	$4.^{\circ}4 \times 4.^{\circ}4$
Size of frame (pixels)	2048×2048
Pixel scale	$7.''73$
CCD operating mode	Full frame
Mechanical shutter	None
Readout channels	4

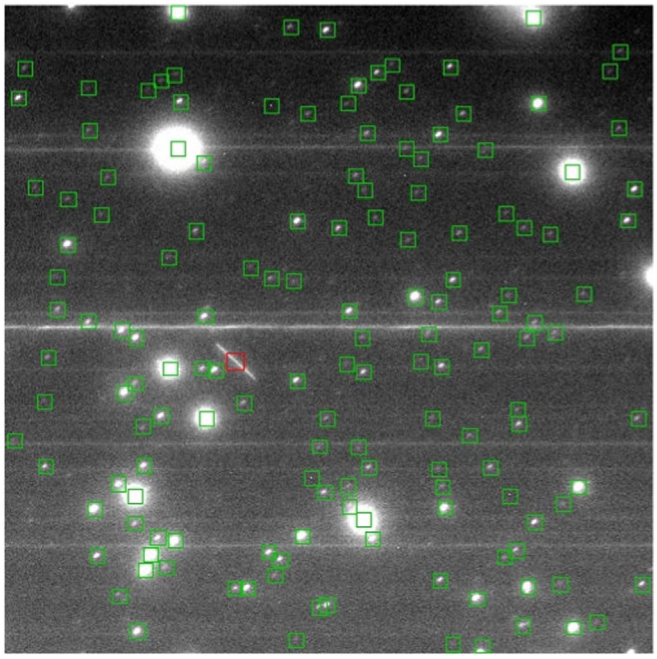

Figure 10. Example of the detection and classification results with real observation image. We find that our framework can correctly detect celestial objects and moving targets. Besides, hot pixels near the bright star in the upper left and linear interference do not have an effect to the final detection results.

Table 3
Improvement Ratio between our Framework and the Classic Framework

Threshold	Improvement Ratio
0.4	1.032
0.5	1.118
0.6	1.250

framework in the Jetson AGX Xavier embedded device provided by Nvidia as the "brain" of WFSATs. The Jetson AGX Xavier has 512-core Volta GPU and can achieve 5 TFLOPs with 16 float point accuracy. Images obtained by WFSATs are transferred into Xavier through its USB 3.1 port and it costs around 0.3 s to process an image with a size of 600 × 600 pixels. For images with a larger size, we can divide them into small images with an overlap area and process them part by part. After the detection and classification step, we cross-match all astronomical targets with different catalogs in the Jetson Xavier. We will broadcast coordinates and types of candidate transients in the whole observation net. Each WFSAT can either use the information as classification and detection results from another neural network and further increase detection and classification ability through ensemble learning, or use these information as guidance to observe a particular sky area.

5. Conclusions and Future Work

Fast and accurate detection and classification of astronomical targets are important for transient observations. In this paper, for images obtained by WFSATs, we propose an object detection and classification framework based on deep neural networks. We compare the performance of our framework and that of the classic framework based on SExtractor with simulated and real observation data. Our framework is robust and has a better performance. We further propose to install our framework in embedded devices to achieve real-time detection and classification ability, which would further increase the observation ability of the whole transient observation net. However, with simulated data, we find that our framework cannot detect all of the targets, which indicates that improvements are still needed for our framework. In our future work, we will optimize the structure of our framework to further improve the performance of our framework.

The authors would like to thank the anonymous referee for comments and suggestions that improved the quality of this manuscript. P.J.would like to thank Dr. Nan Li from National Astronomical Observatories, Dr. Alastair Basden from Durham University, Professor Kaifan Ji from Yunnan Observatory, Dr. Rongyu Sun and Dr. Tinglei Zhu from Purple Mountain Observatory, and Professor Qingfeng Zhang from Jinan University, who provided very helpful suggestions for this paper. All the real observation images used in this paper are annotated by undergraduate students from classes 1701, 1702, 1703, and 1704 of the optoelectronics engineering major and classes 1701 and 1702 from the light source and lighting majors at the College of Physics and Optoelectronics at Taiyuan University of Technology. This work is supported by the Joint Research Fund in Astronomy (U1631133) under cooperative agreement between the NSFC and Chinese Academy of Sciences (CAS), National Natural Science Foundation of China (NSFC)(11503018), Shanxi Province Science Foundation for Youths (201901D211081), Research Project Supported by Shanxi Scholarship Council of China, the Scientific and Technological Innovation Programs of Higher Education Institutions in Shanxi (2019L0225). Complete code can be downloaded from https://doi.org/10.12149/101016.

ORCID iDs

Peng Jia https://orcid.org/0000-0001-6623-0931

References

Bertin, E. 2009, MmSAI, 80, 422
Bertin, E., & Arnouts, S. 1996, A&AS, 117, 393
Bishop, C. M., & Nasrabadi, N. M. 2007, JEI, 16, 049901
Burd, A., Cwiok, M., Czyrkowski, H., et al. 2005, Proc. SPIE, 5948, 59481H
Burke, C., Aleo, P., & Chen, Y.-C. 2019, MNRAS, 490, 3952
Duev, D., Mahabal, A., & Masci, F. 2019a, MNRAS, 489, 3582
Duev, D., Mahabal, A., & Ye, Q. 2019b, MNRAS, 486, 4158
Girshick, R., Donahue, J., Darrell, T., et al. 2013, arXiv:1311.2524
Gonzalez, R. E., Munoz, R. P., & Hernandez, C. A. 2018, A&C, 25, 103
He, K., Gkioxari, G., & Dollar, P. 2017, in 2017 IEEE Int. Conf. Computer Vision (ICCV), ed. L. O'Conner (Piscataway, NJ: IEEE), 2980
Høg, E., Fabricius, C., & Makarov, V. 2000, A&A, 357, 367
Jia, P., Li, X., Li, Z., et al. 2020a, MNRAS, 493, 651
Jia, P., Wu, X., Huang, Y., et al. 2020b, AJ, 159, 183
Jia, P., Zhao, Y., & Xue, G. 2019, AJ, 157, 250
Ketkar, N. (ed.) 2017, Deep Learning with Python (Berkeley, CA: Apress), 195
Kingma, D. P., & Ba, J. 2015, arXiv:1412.6980
Liu, W., Anguelov, D., Erhan, D., Fu, C., & Berg, A. C. 2016, in Computer Vision–ECCV 2016, ed. B. Leibe et al. (Cham: Springer), 21
Li-wen, W., Peng, J., & Dong-mei, C. 2019, ChA&A, 43, 128
Mahabal, A., Rebbapragada, U., & Walters, R. 2019, PASP, 131, 038002
Ping, Y., & Zhang, C. 2017, AdSpR, 60, 907

Ratzloff, J. K., & Law, N. M. 2019, PASP, 131, 075001
Redmon, J., Divvala, S. K., & Girshick, R. B. 2016, in 2016 IEEE Conf. Computer Vision and Pattern Recognition (CVPR), ed. L. O'Conner (Piscataway, NJ: IEEE), 779
Ren, S., He, K., & Girshick, R. B. 2017, ITPAM, 39, 1137
Romano, R., Aragon, C. R., & Ding, C. H. Q. 2006, in 2006 5th Int. Conf. Machine Learning and Applications (ICMLA'06), ed. S. Ceballos (Piscataway, NJ: IEEE), 77
Seiler, M. C., & Seiler, F. A. 1989, Risk Analysis, 9, 415
Sun, R., Yu, S., & Zhao, C. 2019, PASJ, 71, 67
Tachibana, Y., & Miller, A. A. 2018, PASP, 130, 128001
Turpin, D., Ganet, M., Antier, S., et al. 2020, arXiv:2001.03424
Ulyanov, D., Vedaldi, A., & Lempitsky, V. S. 2016, arXiv:1607.08022
Xu, Y., Xin, L., Wang, J., et al. 2020, PASP, 132, 054502
Zackay, B., Ofek, E. O., & Galyam, A. 2016, ApJ, 830, 27
Zhang, S., Zhu, R., & Wang, X. 2019a, arXiv:1901.06651
Zhang, Z., He, T., & Zhang, H. 2019b, arXiv:1902.04103
Zhuang, F., Qi, Z., Duan, K., et al. 2019, arXiv:1911.02685