Monthly Notices
of the
ROYAL ASTRONOMICAL SOCIETY

MNRAS **491,** 5151–5157 (2020)
Advance Access publication 2019 November 28

doi:10.1093/mnras/stz3250

Eliminating artefacts in polarimetric images using deep learning

D. Paranjpye,[1]★ A. Mahabal,[2]★ A. N. Ramaprakash,[3] G. V. Panopoulou,[2] K. Cleary,[2] A. C. S. Readhead,[2] D. Blinov[4] and K. Tassis[4,5]

[1]*Department of Electrical and Computer Engineering, University of Michigan, 1301 Beal Ave, Ann Arbor, MI 48109, USA*
[2]*Division of Physics, Mathematics, and Astronomy, California Institute of Technology, Pasadena, CA 91125, USA*
[3]*Inter-University Center for Astronomy and Astrophysics, Post Bag No. 4, Ganeshkhind, Pune 411007, India*
[4]*Department of Physics and Institute of Theoretical & Computational Physics, University of Crete, Heraklion, GR 71003, Greece*
[5]*Institute of Astrophysics, Foundation for Research and Technology - Hellas, Vassilika Vouton, GR 70013 Heraklion, Greece*

Accepted 2019 November 19. Received 2019 November 9; in original form 2019 October 4

ABSTRACT

Polarization measurements done using Imaging Polarimeters such as the Robotic Polarimeter are very sensitive to the presence of artefacts in images. Artefacts can range from internal reflections in a telescope to satellite trails that could contaminate an area of interest in the image. With the advent of wide-field polarimetry surveys, it is imperative to develop methods that automatically flag artefacts in images. In this paper, we implement a Convolutional Neural Network to identify the most dominant artefacts in the images. We find that our model can successfully classify sources with 98 per cent true positive and 97 per cent true negative rates. Such models, combined with transfer learning, will give us a running start in artefact elimination for near-future surveys like WALOP.

Key words: deep learning – image classication – artefect detection – polarmetry.

1 INTRODUCTION

RoboPol (Ramaprakash et al. 2019) is a four-channel optical polarimeter installed on the 1.3 m telescope at the Skinakas Observatory in Crete, Greece that is primarily used for polarimetry of point sources in the *R* band. Its successor the Wide Area Linear Optical Polarimeter (WALOP) is under development at the Inter-University Center for Astronomy and Astrophysics (IUCAA) in Pune, India. Images contain artefacts resulting from dust patterns, cosmic ray hits, satellite trails, and pixel bleeding contaminating information from celestial objects. With the increasing number of images taken every night from such instruments, it is necessary to automate the analysis of data. However, with humans taken out of the loop it is possible for artefacts to get misidentified as a source and be used in the analysis. This would lead to erroneous results so the detection of such artefacts is imperative.

Early work on detection of artefacts in astronomical images dates to the early 2000s when Storkey et al. (2004) used computer vision techniques such as the Hough Transform to detect linear artefacts like satellite trails, scratches, and diffraction spikes near bright stars. These methods were concerned with detection of linear features and highlighted some of the difficulties of using the Hough Transform when dealing with light-density variations.

Later on the focus shifted to object identification followed by extracting features for the objects and then classification using these features to separate out artefacts with methods like decision trees and random forests (Donalek et al. 2008). Recent years have seen the compilation of terrestrial data sets like ImageNet consisting of a million labelled images with a thousand categories such as human faces, digits, vehicles, flowers, animals etc. Deng et al. (2009) followed by the development of deep learning libraries and models using these data sets e.g. the VGG16 architecture (Simonyan & Zisserman 2014).

With deep learning it is possible to skip the sometimes subjective step of feature extraction and go straight to classification after obtaining a labelled data set (see e.g Cabrera-Vives et al. 2017; Duev et al. 2019a,b). This is at the cost of explainability, but with proper validation and test data sets, the results are still reliable. Additional ways to improve the robustness, and faster convergence using techniques like Mask R-CNN and linear scaling combined with normalization are discussed in some recent papers such as He et al. (2017), Gonzalez, Absil & Van Droogenbroeck (2018), and Burke et al. (2019).

Our task here is to classify objects in RoboPol images into stars and artefacts. RoboPol images contain reflections of bright stars due to the interface between the two Wollaston Prisms used in the instrument and it is this dominant class of artefacts that we target here. The interface between the Wollaston Prisms is shown in the diagram of the optical instrument design described in Ramaprakash et al. (2019), and an example of the reflection artefact in Fig. 1. The green box in the upper left quadrant shows two horizontally extended artefacts separated vertically. A few stars in the vicinity also got included in the box.

★ E-mail: dhruvparanjpye@gmail.com (DP); aam@astro.caltech.edu (AM)

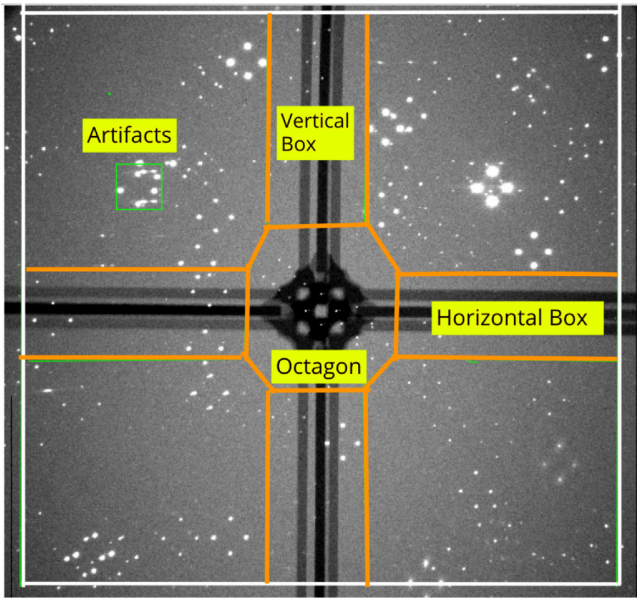


Figure 1. The region within the orange lines describes the restricted area from where we do not extract stars or artefacts. Similarly the regions outside the white lines are restricted areas. Note that the lines are exaggerated only for the purpose of representation.

In this paper, we propose to solve the problem of artefact detection for RoboPol images using an appropriately designed Convolutional Neural Network (CNN). In Section 2, we introduce our approach to the problem of detecting artefacts in RoboPol images. We detail the implementation of our method including pre-processing steps, CNN architecture, and visualization of the output, and in Section 3 we discuss our findings and future possibilities.

2 APPROACH

The RoboPol data base consists of tens of thousands of images taken between 2013 and 2019. We first generate a data set containing stars and artefacts and then develop a CNN to perform the classification. The following is an outline of our method:

(1) Create training data for artefacts and stars from RoboPol images through manual labelling. This includes data for validation and testing. The manual labelling was done by visually inspecting about 100 images and recording the pixel coordinates of the artefacts.
(2) Develop a CNN Architecture tuned through hyperparameter variation.
(3) Train the model using training data obtained in step 1.
(4) Validate the model using validation and testing data.
(5) Implement the model to find artefacts in an arbitrary RoboPol image.

2.1 Training data for artefacts and stars

Reflection artefacts and stars in RoboPol images have x and y extents from several pixels to a few tens of pixels. We chose a size of 64×64 pixels for our cutouts, with the artefacts and stars centred. Each star appears at four locations due to splitting of light from a single source within the instrument, with the locations lying at the vertices of a diamond. A detailed design implementation is available in Ramaprakash et al. (2019).

For each image we generate a catalogue of sources (including stars, reflection artefacts, and any other connected brightness peaks) using Sextractor.[1] We have roughly 10 artefacts and about 250 stars per image. The catalogue comes with flags indicating various conditions such as saturation, proximity to another source, proximity to edge of image etc.[2] About 70 per cent of the visually inspected artefacts had no error. This indicated that relying on just flags is not sufficient to separate artefacts. To obtain training data of stars we make sure that from every image we extract stars of varying brightness and not just from a narrow brightness range. In each image, we chose this range to be one star per magnitude-bin for up to five magnitudes in each image. Likewise, our training data would contain an uniform distribution of magnitudes of brightness and ensure that we are not biasing our neural network by providing training images from a limited magnitude range. We do not use all sources so that the sets of stars and artefacts can stay roughly equal, and hence balanced for the classification process. Fig. 2a and 2b show a sample collection of the training images.

Unlike typical astronomy images, RoboPol images contain a mask (Fig. 1). More details about the mask can be found in Ramaprakash et al. (2019). The code repository and documentation are available at https://github.com/delta-papa/Robopol-artifacts

2.2 CNN architecture

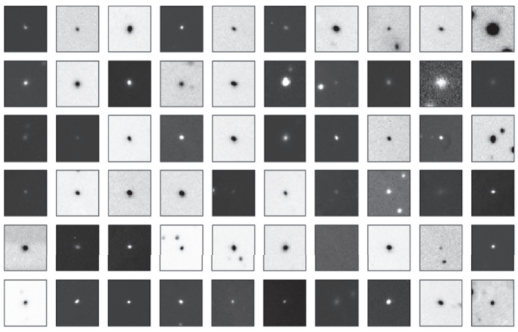
We follow the now standard image classification model developed by the Visual Graphics Group (VGG) at Oxford, UK (Simonyan & Zisserman 2014). Our implementation uses three convolution layers, three max-pooling layers, and two fully connected layers (see Fig. 3). The hidden layers are activated using a ReLU (Rectified Linear Unit) activation. Finally we use a sigmoid activation at the output layer. The first, second, and third convolution layer consist of 32, 64, and 128 filters respectively each with a kernel size of 3×3 and stride length of 1. The max-pooling layers use a kernel size of 2×2 pixels. At the end of the 3rd max-pooling layer we use a dropout layer with a probability of dropping a node as 0.4 for regularization ensuring no single parameter of the neural network has a very high coefficient (Srivastava et al. 2014). The total number of trainable parameters in our configuration are 2 452 993 and we use an Adam optimizer to perform back-propagation (Kingma & Ba 2014). The loss function used is a binary cross-entropy loss.

2.3 Data augmentation and training

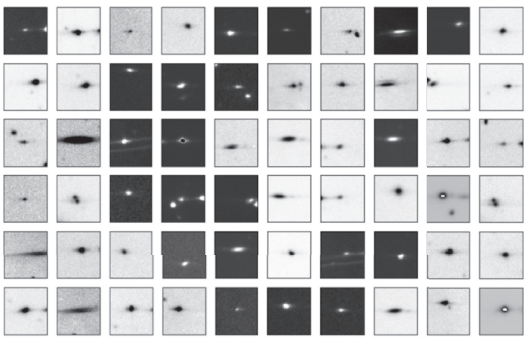
While going through the images we saw that most of the artefacts are due to internal reflections and had a horizontal streak-like shape. For proper training of the neural network we need to generate a large data set of training images. Therefore, to augment the number of images for training, we rotated the cutouts by 180 degrees so that the horizontal nature of the artefacts is preserved.

To split the data into training and validation we used a 80–20 ratio and shuffled the data set randomly while splitting to avoid bias. For training, we had a total of 836 images of stars and 925 images of artefacts. The training data were augmented using the ImageDataGenerator class in the high-level Keras[3] API of PYTHON. We performed horizontal and vertical flipping, width and height shifts and shearing. The shifts were applied to account for possible

[1] https:///sextractor.readthedocs.io
[2] https://sextractor.readthedocs.io/en/latest/Flagging.html
[3] www.keras.io



(a) Cutouts of Stars used as training images.



(b) Cutouts of Artefacts used as training images.

Figure 2. Training images for stars (a) and artefacts (b).

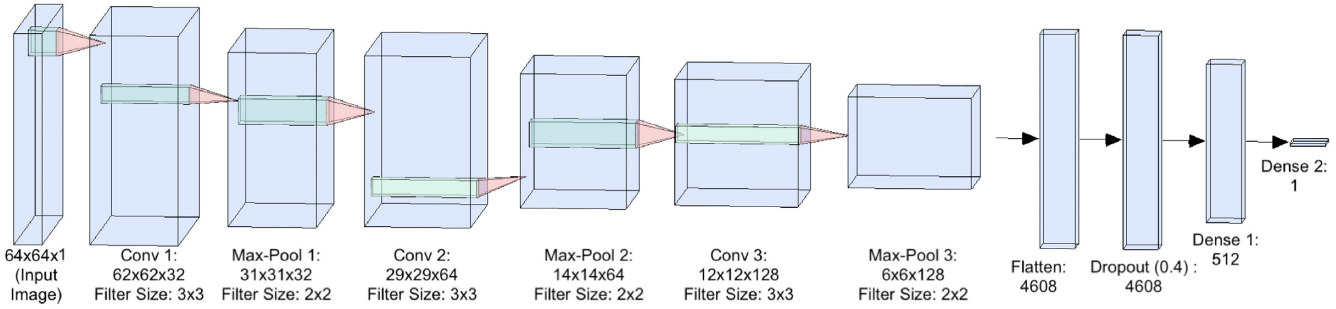


Figure 3. The CNN model consists of three convolutional layers, three max-pooling layers, two fully connected layers, and a dropout layer for regularization. In case of the convolutional and max-pooling layers, the size of the layer and the filter size used are also mentioned. For the dense layers, the total number of nodes are shown. The input is a 64×64 pixel PNG image.

inaccuracies in centring of the samples. The validation and training images were both normalized to [0,1] by dividing by 255, the maximum value of an 8 bit image.

The hardware used for performing training was a 2.3 GHz Intel Core i5 processor and the total training time was 40 min.

2.4 Training performance

The total number of images chosen for training the model was 1408 (80 per cent of the 1761 images), and the remaining 353 images were reserved for validation. Training data were used to update the parameters of the model while validation data were used to only evaluate the model's performance after every update. The batch size used was 4. A total of 100 epochs were used in the training and the steps per epoch was set to 1408/4 i.e. 352. We used a learning rate of 0.001. The training accuracy reached about 95 per cent while the validation accuracy was close to 96 per cent at the end of 100 epochs as seen in Fig. 4.

Besides making small changes to the hyper parameters above, we also implemented a network with two and four convolution layers to see whether there was any advantage in using shallower (two layers) or deeper (four layers) CNNs. The training accuracy and validation accuracy reached about 90 per cent in the shallower network while it reached about 96 per cent in the deeper network. Although training and validation accuracy may be good indicators of the proper working of a CNN in a binary classification problem there are other important parameters we need to consider when the costs of misclassification are high. For example, we are interested in knowing the false positive rate (sources wrongly classified as stars), the false negative rate (sources falsely classified as artefacts),

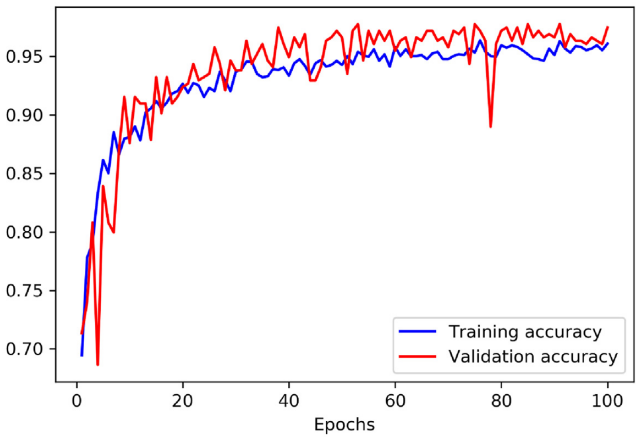


Figure 4. Training and validation accuracy for the final model.

as also the precision (fraction of sources correctly classified) and recall (fraction of stars correctly classified). These numbers are summarized through two metrics viz. F1 score and Matthew's correlation coefficient, and indicate whether our model is working as expected or not.

We need our system to have a high precision and recall score and the F1 score summarizes the two scores by taking their harmonic mean. The Matthew's Correlation Coefficient (MCC) is akin to a correlation coefficient measure between the predicted labels and true labels. A value of +1 indicates perfect positive correlation between the two quantities. Equations (1)–(4) give the formulae for Precision, Recall, F1 score, and MCC, respectively. Note that TP, FP,

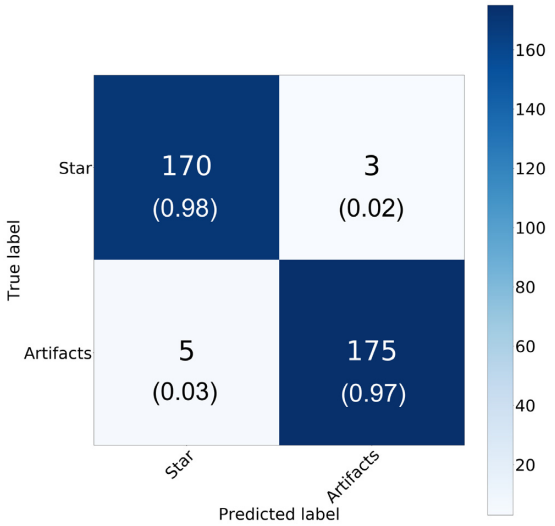


Figure 5. Confusion matrix for the binary classification performed using our model. The normalized numbers are given in brackets. Out of 353 images, 170 were True Positives, 175 were True Negatives, 5 False Positives, and 3 False Negatives. The False Positive Rate is 3 per cent and False Negative Rate is 2 per cent.

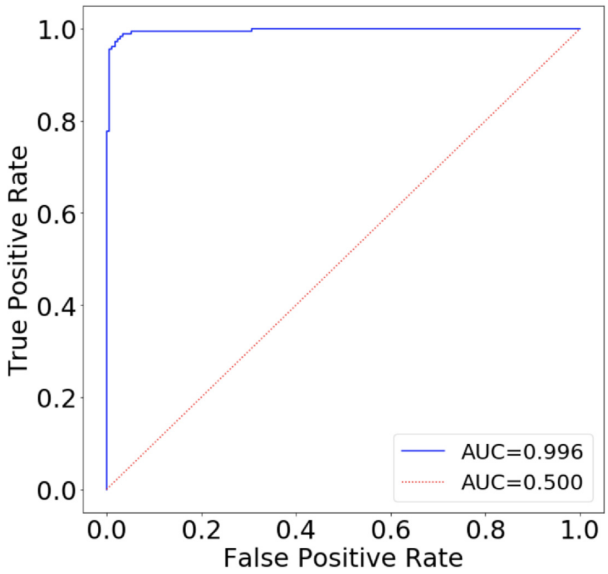


Figure 6. ROC Curve for our final model.

TN, FN, stand for True Positives, False Positives, True Negatives, and False Negatives, respectively.

$$\text{Precision} = \frac{\text{TP}}{\text{TP} + \text{FP}} \tag{1}$$

$$\text{Recall} = \frac{\text{TP}}{\text{TP} + \text{FN}} \tag{2}$$

$$F1\ \text{score} = \frac{2 \cdot \text{precision} \cdot \text{recall}}{\text{precision} + \text{recall}} \tag{3}$$

$$\text{MCC} = \frac{\text{TP} * \text{TN} - \text{FP} * \text{FN}}{\sqrt{(\text{TP} + \text{FP})(\text{TP} + \text{FN})(\text{TN} + \text{FP})(\text{TN} + \text{FN})}}. \tag{4}$$

Figs 5 and 6 show the confusion matrix and ROC curve, respectively. The Confusion Matrix tells us the number of true positives, true negatives, false positives, and false negatives. The Receiver Operating Characteristics (ROC) curve is a measure of the trade-off between the true positive rate and false positive rate for different values of the threshold used in the classifier. The threshold is a value between 0 and 1. If the probability of the source being a star is greater than the value of the threshold, we classify the source as a star else as an artefact. Our threshold is set to 0.5. Ideally, we want the false positive rate to be 0 and true positive rate to be 1. Ideally we expect our ROC curve to have an area (under the curve) to be equal to 1. The Area Under the Curve (AUC) for our model is 0.996 while that for a random classifier is 0.5 as shown by the red dotted line. A zoom-in of Fig. 6 is shown in Fig. 7 which shows the values True Positive Rates and False Positive Rates at different thresholds. It shows that our threshold of 0.5 coincides with 0.75 indicating that our classifier can confidently achieve the same True Positive Rate and False Positive Rate at a higher threshold. Table 1 compares the performance of our model with a shallower and deeper neural network. Based on this comparison, we chose the model with three convolution layers as the model with four layers actually shows a reduction in F1 score and MCC with a deeper and hence computationally more expensive network, possibly due to overfitting. Figs 8 and 9 show some of the false negatives and false positives.

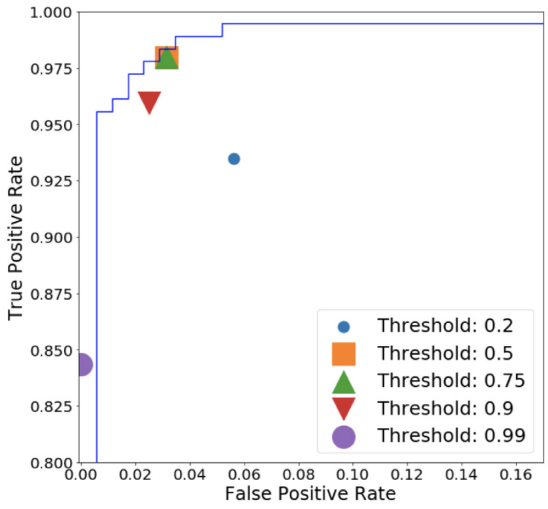


Figure 7. Zoom in of ROC (Fig. 6) to highlight the non-ideal area.

Table 1. Performance comparison of CNN models with different layers. Here FPR is False Positive Rate, FNR is False Negative Rate, MCC is Matthew's Correlation Coefficient.

Parameter	Two layers	Three layers	Four layers
Training accuracy	0.90	0.95	0.90
Validation accuracy	0.91	0.96	0.95
FPR	0.07	0.03	0.02
FNR	0.13	0.02	0.05
Precision	0.88	0.98	0.95
Recall	0.93	0.97	0.98
F1 Score	0.90	0.98	0.96
MCC	0.80	0.95	0.93

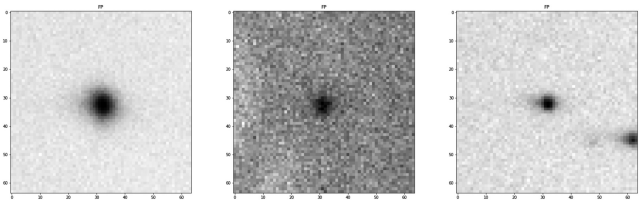


Figure 8. Some of the False Negative classifications where stars were classified as artefacts.

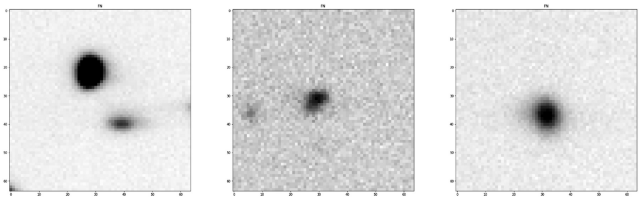


Figure 9. Some of the False Positive Classifications where artefacts were classified as stars. We see that the artefacts here did not have a prominent streak-like shape. Also, the first artefact has a star near the centre and the artefact is slightly away from the centre.

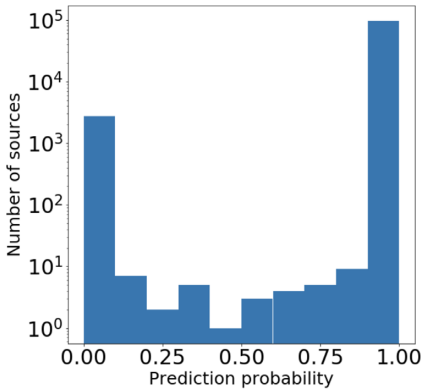


Figure 10. Histogram of prediction probabilities for 91 000 sources.

2.5 Testing and implementation

We used our model to test 100 randomly chosen images – distinct from the training set – from the 40 000 images of RoboPol taken during the years 2013 and 2014. Our goal was to classify all the sources in each of the 100 images into stars and artefacts and analyse the results. For each image, we obtained a list of sources, their positions, instrumental magnitudes, and extraction flag error by using Sextractor. A histogram of the predicted probability of the sources being stars is shown in Fig. 10. Out of 91 000 sources, we have 88 000 sources classified as stars (90 to 100 per cent probability of being a star) and 2500 sources classified as artefacts (0 to 10 per cent probability of being a star). The inset plot shows that in the remaining prediction probability range there are fewer than 10 objects in each bin of size 10 per cent. This means that $\sim$ 0.05 per cent of the sources had probabilities in the range between 10 and 90 per cent.

Figs 11 and 12 show the results of classification on test images. Each image contains a single source with known label. At the top of each image is the probability of the source being a star. Sources in Fig. 11 are artefacts while those in Fig. 12 are stars. In both categories, our classification rates are almost always above 90 per cent.

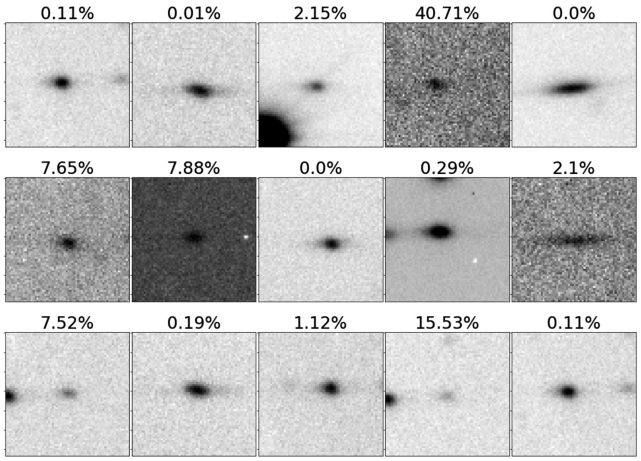


Figure 11. Result of testing the classifier on images of artefacts. All artefacts have been classified correctly with high classification probabilities. The classification probability of a source being a star is shown on the top of each cutout.

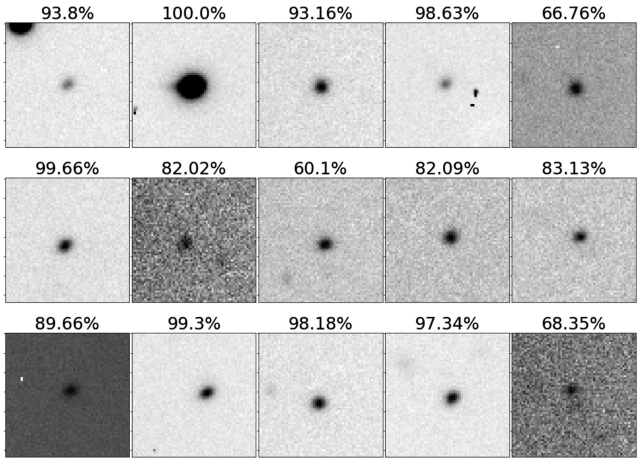


Figure 12. Result of testing the classifier on images of stars. The classification probability of a source being a star is shown on the top of each cutout.

The implementation pipeline takes an input image and the corresponding Sextractor file as its arguments and produces a list of locations of the detected artefacts along with their location marked in the original image with an associated probability. A decision logic diagram is shown in Fig. 13.

2.6 Visualization with saliency maps

A saliency map helps us find the locations of the pixels in input images which need to be changed the least to activate the output filter. This means we find the gradient of the input image with respect to the output score. To visualize a saliency map, positive gradients are chosen that would give us the locations of the pixels activating the output filter. In other words this gives us the location of the object of the relevant class in the input image. A saliency map thus gives us the salient features of the class-specific input image that maximize the class score. A detailed mathematical treatment can be found in Simonyan, Vedaldi & Zisserman (2013). Fig. 14 shows the saliency maps for nine different input images. Observe that for the image in row 3, column 3, only the artefact is visualized

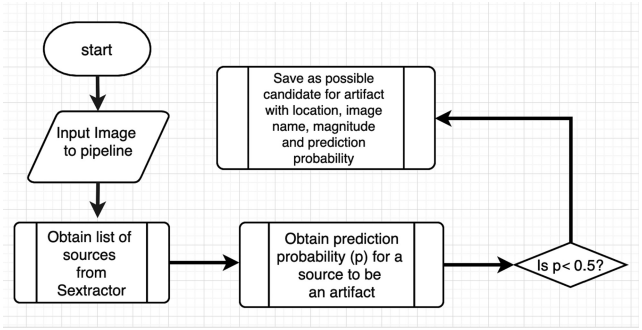


Figure 13. A flow chart showing the decision logic of the pipeline.

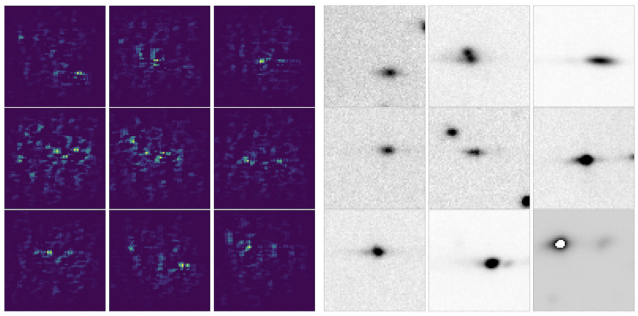


Figure 14. Saliency map visualization (left-hand panel) for nine different input images (right-hand panel) of artefacts. The saliency map shows the positive gradients of the image with respect to the artefact class score and thus the locations of the artefacts in the image. These maps show that indeed the classifier is activated by the artefacts themselves and not their background. Figure plotted using Keras Visualization Toolkit (Kotikalapudi & contributors et al. 2017).

in the saliency and not the star at the top right corner of the image. Sextractor's ellipticity measure alone is not sufficient to separate the artefacts.

3 DISCUSSION

We used convolutional neural networks to solve the problem of detecting artefacts in polarimetric images. Although the use of CNNs in astronomical image classification is not new, this is the first time that they have been used for detecting artefacts in polarimetric images. The efficiency of the method shows its suitability for use in upcoming polarimetry surveys such as the polar areas stellar imaging in polarization high accuracy experiment (Tassis et al. 2018), which will use the novel wide area linear optical polarimeter (WALOP). Our implementation suggests that this method can be reliably used for detecting other kinds of artefacts as well given enough training data. The RoboPol instrument operates down to 16th magnitude in the *R*1 band. Fig. 15 shows that our deep learning model can classify stars down to 15.9 magnitude with a prediction probability better than 0.9. We have also plotted the signal-to-noise ratio (SNR) of the stars on a separate axis. We see that our model works up to SNRs of 15. Thus, our implementation works well with objects within the magnitude range RoboPol observes.

In this paper, we do not use the spatial correlation for stars appearing as a diamond pattern in RoboPol images. That is because the diamond structure in each image of RoboPol is specific to the RoboPol polarimeter design and would not be present in a single image of future polarimeters such as WALOP.

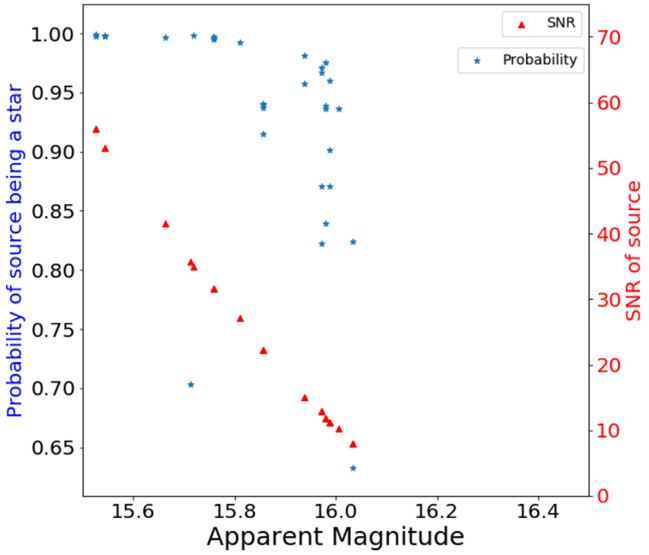


Figure 15. Scatter plot of prediction probability and signal to noise ratio (SNR) of visually inspected stars from a single image versus the apparent R1 magnitude. The 42 stars span a magnitude range from 12 to 16.

In the RoboPol data set, we had majority of artefacts due to scattering of light from off-axis stars at the interface of the Wollaston prisms (Ramaprakash et al. 2019). Our method demonstrates that a binary classifier trained on images of stars and artefacts can successfully differentiate between them. Our training data does not contain enough examples for artefacts such as satellite trails or bleeding pixels and as a result deep learning them is non-trivial without aggressive data augmentation. There already exist methods to remove such artefacts. Out-of-distribution detection networks (Huang et al. 2019) can also be used to detect such infrequent outliers. The final pipeline can incorporate such methods to deliver artefact-free products.

ACKNOWLEDGEMENTS

The work has been funded by the National Science Foundation under the NSF grant (161547). AM acknowledges support from the NSF (1640818, AST-1815034) and IUSSTF (JC-001/2017). KT acknowledges support from the European Research Council under the European Union's Horizon 2020 research and innovation program, under grant agreement no. 771282.

REFERENCES

Burke C. J., Aleo P. D., Chen Y.-C., Liu X., Peterson J. R., Sembroski G. H., Lin J. Y.-Y., 2019, MNRAS, 490, 3952
Cabrera-Vives G., Reyes I., Förster F., Estévez P. A., Maureira J.-C., 2017, ApJ, 836, 97
Deng J., Dong W., Socher R., Li L.-J., Li K., Fei-Fei L., 2009, 2009 IEEE conference on computer vision and pattern recognition. IEEE, Miami, FL, p. 248
Donalek C., Mahabal A., Djorgovski S., Marney S., Drake A., Glikman E., Graham M., Williams R., 2008, in AIP Conf. Proc. Vol. 1082, New Approaches to Object Classification in Synoptic Sky Surveys. Am. Inst. Phys., New York, p. 252
Duev D. A. et al., 2019a, MNRAS, 482, 2039
Duev D. A. et al., 2019b, MNRAS, 486, 4158
Gonzalez C. G., Absil O., Van Droogenbroeck M., 2018, A&A, 613, A71

He K., Gkioxari G., Dollár P., Girshick R. B., 2017, IEEE Conference on Computer Vision (ICCV), 2980
Huang Y., Dai S., Nguyen T., Baraniuk R. G., Anandkumar A., 2019, preprint (arXiv:1907.04572)
Kingma D. P., Ba J., 2014, 3rd International Conference for Learning Representations, San Diego
Kotikalapudi R. et al., 2017, keras-vis. Available at: https://github.com/raghakot/keras-vis
Ramaprakash A. N. et al., 2019, MNRAS, 485, 2355
Simonyan K., Zisserman A., 2014, preprint (arXiv:1409.1556)
Simonyan K., Vedaldi A., Zisserman A., 2013, CoRR, abs/1312.6034
Srivastava N., Hinton G., Krizhevsky A., Sutskever I., Salakhutdinov R., 2014, J. Mach. Learn. Res., 15, 1929
Storkey A. J., Hambly N. C., Williams C. K. I., Mann R. G., 2004, MNRAS, 347, 36
Tassis K. et al., 2018, preprint (arXiv:1810.05652)

This paper has been typeset from a TEX/LATEX file prepared by the author.