An Investigation into Deep Learning Techniques to Vet PHT Candidates

Candidate Number: 1025032 Project Number: AS23

Supervisors:
Professor S. Aigrain
Dr Oscar Barragán

09/05/2021

Abstract

This project trials the method of using a Convolutional Neural Network to vet the transits identified by Planethunters-TESS, a Citizen Science initiative where participants sift through data produced by the TESS satellite to identify potential exoplanets. In existing literature, there are two prevailing methods used to create a training set: using human vetting of real data, or using simulated data. This project trialled a new method of using automation to label real data. The benefits of this method are that no manual labour is required to vet the data, and that results are available much more rapidly. The resulting training set led to a Neural Network with a high accuracy, and an average precision of 53.14% that is comparable to those created using other methods. This report demonstrates how a Neural Network trained with an automatically labelled dataset can perform en par with existing methods.

1 Introduction

Over the past two decades, our ability to detect exoplanets has developed immensely due to the improvement of data collection and identification methods. We are now able to collect data of a volume which is beyond manageable manually. Therefore multiple efforts have been made to automate the classification of signals derived from these methods to both accelerate the process and correct for blind spots in previous automatic methods.

1.1 Exoplanet Detection Methods

There are many methods to identify exoplanets, however this report focuses on *Transit Photometry* as this is the method used by TESS. The first exoplanet discovered through the Transit Photometry method was OGLE-TR-56-b in Konacki et al. (2003) and the number of new discoveries made using this method has grown since then. Other methods include the Radial Velocity (RV) method (Murray and Correia, 2010) that was first used to discover 51 Pegasi b (Mayor and Queloz, 1995), and Direct Imaging that was first demonstrated for discovery in Chauvin et al. (2004). The RV method is used to support the TESS mission by validating exoplanet candidates identified through the Transit Photometry method with follow-up observations. This method is typically performed by ground-based telescope arrays, such as the VLT, using instruments like ESPRESSO (Pepe et al., 2010).

Transit Photometry is performed by examining the light flux originating from a target star. The flux is recorded by a telescope (space or ground-based) to produce a time series. Exoplanets orbiting this host star are then identified through observations of dips in the flux. This may occur when the planet either transits (passes in front), or undergoes an occultation by (passes behind) its star. Transiting planets directly block the light coming from the star. Key to the understanding of occultation dips is the fact that, when we observe a target star, we not only record the light coming directly from it, but also the light that is reflected off the planet's day side. When a planet is occulted, a dip occurs because the planet's day side is no longer visible nor able to reflect light from its parent star to the observer. Transit dips have a significantly larger magnitude than dips caused by occultations; for this reason, occultations are only used for the verification of exoplanets. The periodic pattern of dips is shown in Figure 1 and is further explained in Winn (2010).

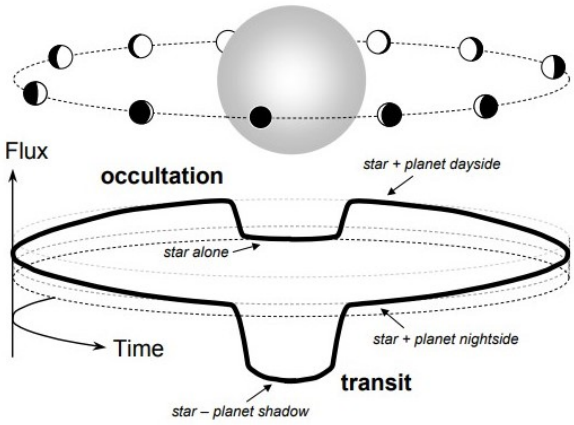


Figure 1: A graphical depiction of dips caused by occlusions, including a plot of combined flux from the planet and host star produced in Winn (2010)

There are however limitations to the transit method. Primarily, a planet has to have an orbit in a specific plane relative to the observer, one that transits the star. The higher the relative inclination (the nearer the exoplanet transits the star's edge), the harder it becomes to identify these planets as the dips become less stark. This can be quantified by this formula for the probability for a transit to be observable given a star's radius R_*, a planet's radius R_p, the eccentricity of the planet's orbit e, the semi-major axis a and the observer's stellar longitude ω.

$$p_{tra}^{(+)} = p_{occ}^{(-)} = \left(\frac{R_* + R_p}{a}\right)\left(\frac{1 \pm e\sin\omega}{1-e^2}\right)$$

1.2 Space-Based Transit Searches

Although the first exoplanets identified through Transit Photometry were identified by ground-based telescopes, this method is much more successful without interference from the atmostphere. Therefore, several space telescopes have since been launched to identify exoplanets using the Transit Photometry method. The first mission was CoRoT (Auvergne et al., 2009), launched in 2007, with dual aims of studying astroseismology and planet search. The Kepler space telescope (Koch et al., 2004), launched in 2009, was the first one of its kind to be developed with the sole purpose of detecting exoplanets. Since that mission, we have seen the transition to the K2 mission (Howell et al., 2014), the launch of TESS (Ricker et al., 2014), and the preparations for PLATO (Rauer et al., 2014).

Kepler (Koch et al., 2004) was a great leap forward in the exoplanet detection field, with its improved specifications and length of continuous observation time. After the unfortunate loss of two of the telecsope's reaction wheels, it was repurposed for the K2 mission (Howell et al., 2014), which also yielded brilliant results. These two missions lead to the discovery of 2,662 exoplanets in total, orders of magnitude more than previous missions. Alongside these discoveries, there was significant development of the surrounding scientific infrastructure. This included the Kepler Science Processing Pipeline (Jenkins et al., 2010) and novel methods of signature detection, such as the Machine Learning (ML) methods inspiring this work (McCauliff et al., 2015) (Shallue and Vanderburg, 2018). These methods, introduced near the end of Kepler's mission time, managed to identify a few planets that previously eluded human vetters. All these developments lay the groundwork for TESS (the Transiting Exoplanet Survey Satellite) to become the success it is today (Ricker et al., 2014).

By 2026, we aim to have launched the next generation of exoplanet hunters called PLATO (Rauer et al., 2014). The pipeline for PLATO is planned to contain Machine Learning elements for the first time and will directly deliver exoplanet candidates rather than PDF summaries of signals.

1.3 The TESS Mission

TESS will be observing significantly more of the sky than previous missions, and it aims to perform a full sky observation. TESS will observe approximately 85% of the sky, whereas Kepler only targeted a fraction of a percent with a fixed view. TESS targets stars much brighter (10-100x) than those that Kepler targeted, this is due to the lower pixel resolution of its cameras and increased background signal from its orbit. These design compromises have led to much noisier data, however the fact that its target area is so much larger means that TESS is able to observe many more systems ($\sim 3x$). Every target star is assigned a TESS Input Catalog (TIC) ID - how these are selected is outlined in Stassun et al. (2018).

TESS will split the sky up into *sectors* measuring $24° x 96°$, observing each sector for two of its orbits (27.4 days). These sectors are then split further into a 1x4 array across the 4 cameras of TESS. Due to some sectors overlapping, certain sections of sky will receive much more than this duration of observation. TESS collects the majority of its data at a 2 minute sampling rate, whereas Kepler had a shorter observation cadence, meaning TESS produces less precise measurements.

TESS performs a data down-link manoeuvre once every orbit at perigee. These observations from TESS are fed into various discovery pipelines, the primary one being the TESS Science Processing Operations Center (SPOC) (Jenkins et al., 2016). The SPOC pipeline analyses signals received from TESS with numerical methods and labels signals that surpass a certain signal-to-noise ratio as a Threshold Crossing Event (TCE). These TCEs are the main focus of most identification efforts, however SPOC often fails to pick up single transit signals due to its reliance on finding periodic patterns for TCEs. If any signal observed by TESS is believed to be worth further observation - i.e. believed to be an exoplanet - then it will be added to the TESS Objects of Interest (TOI) catalog (Guerrero et al., 2021).

Although most identification efforts rely on the TCEs identified by SPOC, it is worth noting they are not always good exoplanet candidates due to the numerical nature of their identification. These TCEs can be further classified in several ways; this report uses three classifications previously used in Shallue and Vanderburg (2018).

First, there are observations that contain no useful information, and these are referred to as a Non-Transiting Phenomenon (NTP). These are relatively easy to identify due to the fact that the noise is typically systematic. Second, some observations appear similarly to transiting planets, but are not in fact caused by one. Reasons for this may be stellar variability (natural changes in the amount of light a star emits) or eclipsing binary systems. A star transiting another star can appear similar to a planet transiting its parent star, although the change in light flux received is often much sharper and more sudden. This subtle difference allows the trained eye to distinguish binary systems from a transiting planet. These observations are referred to as an Astrophysical False-Positive (AFP) and make up the majority of signals. Finally, a few observations may seem to display the signature of a planet, but typically these will require follow-up observations to be confirmed. For this reason, we refer to an observation of this classification as a Planetary Candidate (PC).

Identifying PCs from TCEs follows a two stage process. *Triage* occurs first, where observations are labelled as an NTP if applicable. Remaining observations go through *vetting*, where they are labelled as either an AFP or a PC. Vetting is a difficult process for a trained eye, let alone an algorithm, due to the two classifications sharing many characteristics.

1.4 Transit Candidate Vetting

A range of attempts have been made to automate the process of vetting and triage for both Kepler and TESS data. However, the vast majority of current vetting is still manual. This is performed by numerous teams working with automatically produced PDF summaries for each TCE. Many of the automation attempts have made use of Machine Learning methods; in fact, the official Kepler team investigated the use of *Autovetter*, although later decided not to go ahead with it (Catanzarite, J. H., 2015).

Those working on automating the analysis of the Kepler dataset saw success for both vetting and triage. In fact, many of these approaches led to the discovery of new exoplanets (Shallue and Vanderburg, 2018) (Dattilo et al., 2019). For TESS, on the other hand, there was some success for triage (Yu et al., 2019), but useful vetting attempts seem to have eluded the community as of yet.

The vast majority of automation attempts so far have either required manual vetting of a training set or have used simulated data. Manual vetting is a long and arduous task, especially for the large datasets required for Neural Network approaches. Even with significant man-hours invested, it still may not yield a successful result. The other approach of using simulated data often does not abstract well for use in real life scenarios, due to the increased number of data artifacts throughout the set. Therefore, I have taken the uncommon approach of attempting to create a dataset automatically from real data to avoid the need for human vetting.

1.4.1 First Attempts

Some early attempts at this classification problem used a Random Forest Algorithm (McCauliff et al., 2015). This worked by extracting a set of features from each light curve (a time series of the light flux received by TESS). Then, each set of features would be evaluated across a large number of randomly constructed decision trees (hence, Random Forest). Finally, a majority vote would be taken across all the trees to classify the light curve.

A variation of this method was explored in Reis et al. (2019), called the Probabilistic Random Forest algorithm. This algorithm was designed with data uncertainty in mind, as most ML algorithms ignore this problem and use datasets with a high signal-to-noise ratio (SNR). This was the algorithm used by the previous MPhys candidate with the SPIOx group (Anon. CN: 1015761, 2020).

However, the field quickly moved on to using Neural Networks. More specifically, they did so with Convolutional Neural Networks (CNNs) becoming more common place. This was mainly because, for the K2 dataset, CNNs typically performed better than the Random Forest method. The first example of this can be found in Shallue and Vanderburg (2018), and most of the papers that followed were strongly influenced by their work, including everything from the dataset creation to the Neural Network's architecture.

1.4.2 Recent Developments

In more recent years, we have seen the introduction of scientific domain parameters in Ansdell et al. (2018), increasing overall accuracy from 95.8% to 97.5% and precision from 95.5% to 98.0%. The first papers for TESS focused on using simulated data to train models for vetting and triage (Osborn et al., 2020). This yielded very successful results when tested against simulated data, with an average precision of 97.3% for a binary classification model. However, they only achieved an indication that the model could be transferable when applied to real data, recovering 61% of the TOI catalog existing at that time.

An alternative method was used in Yu et al. (2019), where an algorithm was trained using real TESS data. The training set was prepared by manually vetting a list of TCEs from sectors 1 through 5. Yu et al. (2019) used a similar architecture to the rest of the literature for triage, achieving an average precision of 97.0%. For vetting, they used a variety of models, but the average precision was significantly lower for all. The architecture most similar to those used in this report achieved an average precision of 60.5%, however a more complex architecture did achieve a score of 69.3%.

1.5 Planethunters-TESS

SPOC relies on periodic patterns to identify TCEs, this means it will often miss promising signals with only a single transit in one observation. Planethunters-TESS (PHT) (Eisner et al., 2020) is a citizen science project aiming to assist in the identification of exoplanets that undertook such a transit.

Due to the sheer volume of data TESS produces, it is barely possible for dedicated teams to sift through the TCEs identified by SPOC, let alone the data that might contain single transit observations not picked up by SPOC. Therefore, the premise of PHT is to outsource the initial triage to citizen scientists (over 22,000 of them) by training volunteers to identify PCs from the raw data - in effect, bypassing SPOC. As of 2021, PHT has found many promising PCs, 90 of which are presented in Eisner et al. (2021). Not only this, but 73 of these PCs have only had a single transit observed by TESS, which would typically have been missed by SPOC's labelling process.

1.6 Objectives of this Work

The initial objective of this report was to aid the PHT team to vet results from their project. Given the quantity of data TESS produces, and the large number of citizen scientists contributing to the project, the data produced by PHT is currently too vast to manually vet.

Previously, work had been carried out by another MPhys student that used a Probabilistic Random Forest trained on TCE data (Anon. CN: 1015761, 2020). Unfortunately, the method did not abstract well to the PHT use case, with more general light curves and no supporting data. Therefore, this report takes another approach of using a Convolutional Neural Network (CNN) for classification instead. We believed that this approach may abstract better from the defined TCEs to the more general PHT use case. Unfortunately, I could not achieve a high enough degree of precision with the initial TCEs for the abstraction to be justified.

Along this path of experimentation, I established a secondary objective to examine the automatic creation of a dataset, as I noticed that many papers were having to invest significant time into manually vetting their datasets.

2 Data Set

Machine Learning projects are only as accurate as the datasets they use. Hence, it is key to the understanding of this project to examine how the datasets were created and the variations between them. I will outline the method I followed to gather and process the different sections of the data. Alongside this, I will explain how I automated the labelling of the dataset.

2.1 Data Sources

The data used for this project originated from a variety of sources. It is primarily sourced from the Mikulski Archive for Space Telescopes (MAST). I used cURL to download a list of TCEs per sector in csv format (MAST, 2020) and the TOI catalog (as of 30/12/2020). I already had local access to all raw time series files produced for each TIC target, as they had been downloaded prior for PHT.

The project also used data validation files, which are secondary data products of TESS produced for each TCE. These files provide parameters and a light curve detrended from systematic noise. To do so, I wrote a couple of programs in Python that looped through all the TCEs in each sector list and obtained the appropriate data. A full list of the software packages used can be found in Appendix A and the code can be found in this GitHub repository (Anon. CN: 1025032, 2021).

2.2 Time Series

I used a method that was first outlined in Osborn et al. (2020) to process each TCE. For this method, I extracted a time series for both the light flux received and the centroid displacement from the target star position. A centroid is the flux-weighted mean position of the light received from a target star. Alongside this, I extracted an assortment of both stellar and orbital parameters that best characterised the TCE.

The time series of light flux is key to identifying when transits occur. The centroids allow for better vetting procedures as they undergo significant movement for AFPs. The scientific domain parameters allow for easier exclusion of NTPs and AFPs due to extreme values derived from incorrect fits.

2.2.1 Extraction

I extracted the time series of the light flux for each TCE from the data validation file. I used the *LC_DETREND* column, which has been processed to remove any obvious instrumental noise. The data for centroid displacement was found in the raw time series files. I chose the momentum-based measurements, as opposed to the more accuracte point-spread function measurements, due to the fact that a significant number of files only contained the prior.
To extract centroid information, I first took the displacement for each axis from the columns *MOM_CENTR1* and *MOM_CENTR2*. Then, to remove instrumental noise, I filtered them by the *QUALITY* column. Finally, I added the two axes measurements in quadrature to attain a radial displacement time series.

2.2.2 Processing

The next step was to process these time series to be understandable by the Neural Network. I began to do so by extracting the following transit parameters: epoch, orbital period, transit duration and detected depth (difference in flux received in and out of transit). I then phase-folded both time series, centering the transit in the process. Following this, I median binned the data into 2001 bins for a *global view*; this view shows the behaviour across the entire *orbit's duration*. I then also produced a *local view* with 201 bins, which was more focused around the transit – more specifically, it showed a section of the light curve four *transit durations* either side of the centre of the transit.

To normalise the light curve, I subtracted the out-of-transit median and divided by the detected depth of the transit. For the centroid time series, I again subtracted the out-of-transit median, but this time divided by the standard deviation of the series to normalise it.

I corrected each series for NaNs found throughout the time series. Often, large chunks of missing data in the original time series led to empty bins after processing, therefore creating gaps in the resulting views. If over 50% of data was missing I would discard the TCE, otherwise I would linearly interpolate across NaNs.

Finally I would store the local and global views in separate *npy* files. These files significantly reduce the time required to load data and therefore optimised time taken to train each model.

2.2.3 Examples

A processed light curve representing a PC will look similar to the data plotted in Figure 2 and the corresponding centroid might look similar to that in Figure 3. More in depth examples can be found in Appendix B.

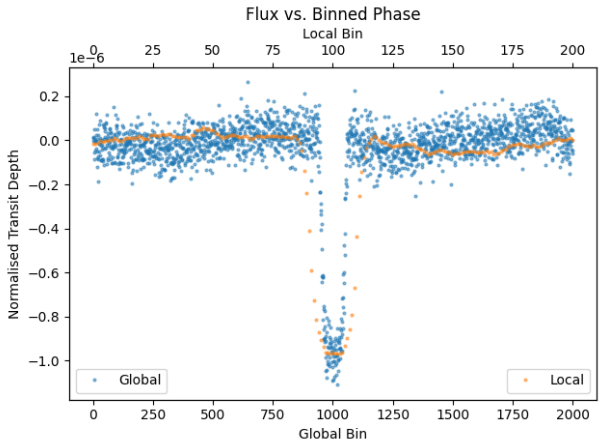


Figure 2: Processed LC of TIC 104024556 TOI 748

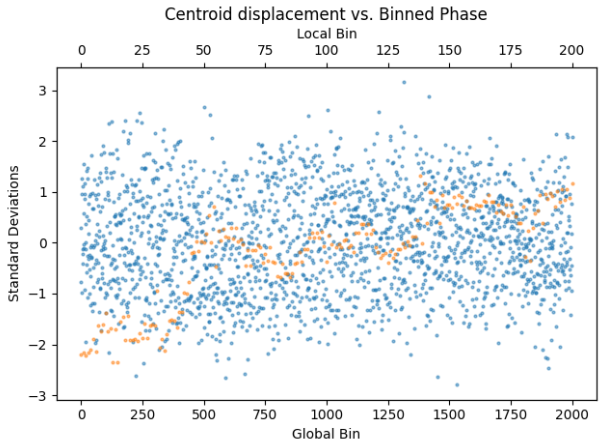


Figure 3: Processed Centroid of TIC 104024556 TOI 748

2.3 Scientific Domain Parameter Processing

I used a pre-established set of Scientific Domain Parameters (Osborn et al., 2020) and extracted these values from a variety of data files listed here:
- Semi major axis scaled to stellar radius, expected transit count, transit depth, ingress duration, impact parameter, signal to noise ratio, ratio of MES to expected MES, ratio of planet radius to star radius and log(ratio of planet radius to earth radius) were all found in the sector's list of TCEs.
- Band magnitude, stellar radius, total proper motion, log of stellar g, stellar melaticity and effective temperature were all found in the TCE data validation file with the light curves.

A few of these values required further processing but all of the data was sourced from these two locations. Earlier on, I experimented with attaining some additional parameters from the MAST API but this meant that creating the training set took far longer due to excessive API requests.

Despite this process, data was still missing under a few of the headings. I set these to sentinel values of 0 post normalisation, as the median value would be equivalent to a negative case due to negative cases vastly outnumbering positive cases in this dataset. Finally, I normalised all values to be between 0 and 1.

2.4 Labelling Automation

The next step was to label each entry in my dataset as either a PC or not. For an MPhys project, it was unrealistic to find a team to vet several thousand light curves, so I made the assumption that any TCE that had not made it on to the TOI list was not a PC and one that had was (i.e. the TOI list was exhaustive). This is a significant assumption, but was necessary to limit the project's scope.

Alongside this, there is no 1:1 pairing of TCEs to TOIs; each TOI is paired to a TIC ID, and each TIC ID can be associated with several TCEs. Each TOI also has several established scientific domain parameters stored with it. Therefore, I was able to match TCEs to TOIs by comparing the parameters of each TOI to the parameters of every TCE associated with it. However, as TOIs are manually recorded, there is a great deal of inconsistency in the values recorded. Some entries use bankers' rounding (typically used by most programming languages), whereas some entries use standard rounding (probably done manually), and to inconsistent degrees of accuracy. For this reason, I analysed the datasets by matching several features across varying degrees of accuracy which has been visualised in the results section.

2.5 Sector Variation

Sector 12 was particularly noisy and did not create a useful dataset when it was run through my program for creating a training set, and so I cut it from the final training set. I encountered difficulties when accessing the raw files for sector 16, as they were stored in a different way to the other sectors. I had enough data within other sectors to justify dropping this sector too.

There was significant variation observed across the sectors; some sectors had more TCEs identified than others, and some sectors had more PCs. However, when labelling, I made the approximation that all sectors would have a similar proportion of TCEs being identified as TOIs. Although it is not expected for there to be similar amounts of exoplanets across different sectors, it is reasonable to assume a similar ratio of PC, AFP, and NTP classifications within the TCEs generated for each sector.

3 Machine Learning Methods

In the broadest sense, there are two approaches to Machine Learning: supervised and unsupervised learning. Supervised learning is where an algorithm is directed and trained using a labelled dataset, whereas unsupervised learning is where the algorithm defines its own labels through differences that it observes. The prior MPhys candidate, the general literature and this report all have so far approached this problem with supervised learning, using a variety of algorithms.

As addressed earlier, there are many potential approaches for Machine Learning Classification. The approach used in this report, Convolutional Neural Networks (CNNs), was first established in this domain in Shallue and Vanderburg (2018) and has since been developed in multiple papers, primarily by two groups Dattilo et al. (2019) and Ansdell et al. (2018). The general architecture of the Neural Network has remained the same across all these papers.

3.1 Deep Learning Models

There are two kinds of Neural Networks: those that are Fully Connected (FCNN) and those that are Convolutional (CNN). A generalised Neural Network takes an array of scalar input values and places each in a *neuron*. These values could represent anything from the pixels of an image to the elements of a time series, such as in this scenario. Each set of neurons is called a *layer*, and several of these layers will be used to construct a Neural Network. The first is called the *input layer*, the last is the *output layer*, and anything in between is called a *hidden layer*. Hidden layers are so called because of their obfuscation from the developer and it is often very hard to derive any meaning from these layers.

Whether a network is Fully Connected or Convolutional is determined by how its layers are connected to one another. Due to this, networks can be comprised of both Fully Connected and Convolutional layers, and these are also referred to as CNNs. For an FCNN, each neuron in a layer is connected to every neuron in the previous layer. On the other hand, in a CNN, each neuron is only connected to a small subset of neurons in the previous layer. These connections are simply a multiplied weighting of the values of the neurons in the previous layer, to produce a new scalar value in the neuron of the subsequent layer. The way these connections are formed is called an *activation function*. In my model, and most of the literature, a ReLU activation function was used (Ramachandran et al., 2017). A ReLU function simply returns $ReLU(z) = max(0, z)$

CNNs have an additional type of layer, called a *pooling layer*. The purpose of these layers is to "widen the view" of each neuron. Initially, each neuron might be only handling a single pixel each. After one pooling layer this might be a col-

lection of pixels, and then after another it could be a section of an image. This allows CNNs to understand spatial resolution on a number of scales, something that is unachievable with an FCNN. It is worth noting that this spatial resolution means that a CNN can understand a sense of order when examining a time series as well. Again, there is variation in exactly how this pooling operation is performed. My model and the literature use *maximum pooling*, where each neuron takes the maximum value from a defined set of neighbouring neurons as its own.

Finally, there will be an output layer, which consists of a single neuron, and its connections to the previous layer will always be fully connected. In my model, the activation function used for the output layer is a Sigmoid activation function (Han and Moraga, 1995). This activation function is particularly useful as it outputs a value between 0 and 1, and so the output can be viewed as a probability.

3.2 Training Process

The training process for any neural network is relatively simple. First, a number of models with randomly weighted connections will be produced and automatically tested against the training set data. This testing will be performed by a *loss function*, which determines how far from the ground truth the model actually is. The weightings of the model will then undergo modifications influenced by an *optimisation function* that takes the loss as its input. The model is subsequently tested again and this process repeats for a specified number of *epochs*. For my model, I used 50 epochs, the Binary Cross-Entropy Loss (BCELoss) function (Liu and Qi, 2017) and the Adam optimisation function (Kingma and Ba, 2014). The Adam optimisation function was a good choice due to its ability to manage more noisy datasets.

Alongside this, it is worth noting that the training can be further divided into *batches* to optimise memory load. Rather than processing the entire dataset in one go, one can split the dataset into a number of batches. Then one can process a batch, modify the model with the above cycle and repeat this until the dataset is exhausted. Completely cycling through a dataset is called an epoch when using batches, and at this point the process can start over. Although this typically has the disadvantage of creating a less accurate model than one trained with the full dataset at once, it makes the computing requirement much more reasonable. I tried using batches of both 128 and 256 time series but, as expected, saw better performance from the 256 batch model.

Finally, the performance of the model is validated using data it has not seen before. This is often done by dividing the initial dataset into a *training set* and a *validation set*. I decided to use 10% of my initial dataset for validation purposes, and the light curves used for validation were randomly selected using a simple Python script.

3.3 Model Architecture

I began by cloning the Git repository of Exonet and built my model from there. My code is largely similar to Ansdell et al. (2018) and great credit goes to them for making their code publicly available.

I wrote my own DataLoader for the program, as I created my training set in a different manner to the original Exonet paper, and so how it was stored, down to the actual file types, was different.

The next thing to address was how to balance the dataset. This means removing any bias the network might pick up from the probability of classifications appearing in the dataset. As the number of positive cases in my training set were massively outweighed by negative cases, the network would always predict negatively if this was not accounted for in some way. There are two ways to fix this. The first is to load a subset of the total dataset that is balanced (equal amounts of positive and negative cases) for each epoch. The alternative option, and the method used in this report, is to weight the model. This method was more ideal because PCs were very underrepresented in the training set, and hence a subset approach would mean a large amount of negative cases would go unused. Exonet used the subset approach because their dataset was far more balanced than mine, meaning it was necessary to add weighting to this code. This was done by calculating the number of positive and negative cases for each batch, and then weighting the loss function of each class by the proportion of the opposite class in the batch. I initially tried weighting by the proportions of each class throughout the entire dataset, however I found that weighting per batch yielded significantly better results.

The final architecture of the model can be found in detail in Appendix C and the code on GitHub (Anon. CN: 1025032, 2021). As illustrated in the results section, there were a few iterations of this model. Sometimes scientific domain parameters were excluded, and sometimes centroids were excluded.

4 Results

4.1 Training Set Summary

The final training sets produced displayed a variation between sectors that made automatic labelling more difficult. Specifically, sectors 14-19 produced TOIs that were much harder to assign accurately to their respective TCEs than the other sectors. Figure 4 and Figure 5 are designed to display the difficulty in assigning labels confidently and highlight the variation between sectors. In Figure 4, a feature is said to be shared between a TOI and a TCE if there is no more than a 10% difference. In Figure 5, this threshold is set to 1%. The colour of each segment of the bars in these figures represents the number of features a TOI and TCE must share to be counted as one and the same. There are 7 colours present on the bar chart representing 1-7 features matched. The further up the bar, the fewer features must be shared between a TOI and TCE to be declared as the same. For example, reading from Figure 4, one can see that, in sector 5, if three shared features (at 10% confidence) are required to match a TCE to a TOI, approximately 60 positive cases would be found. Whereas, if we only require one, approximately 90 positive cases would be found.

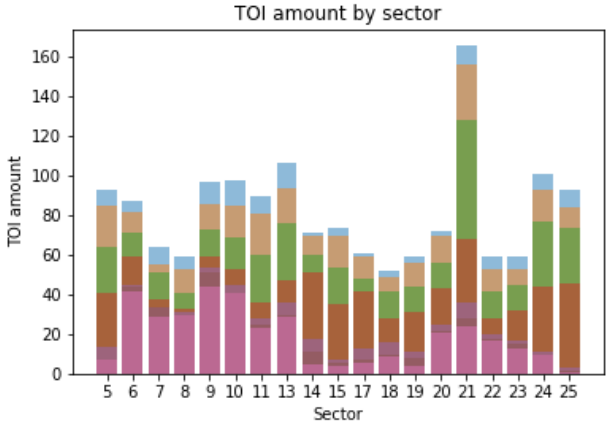


Figure 4: TOI-TCE match *amounts* by sector at 10% feature similarity

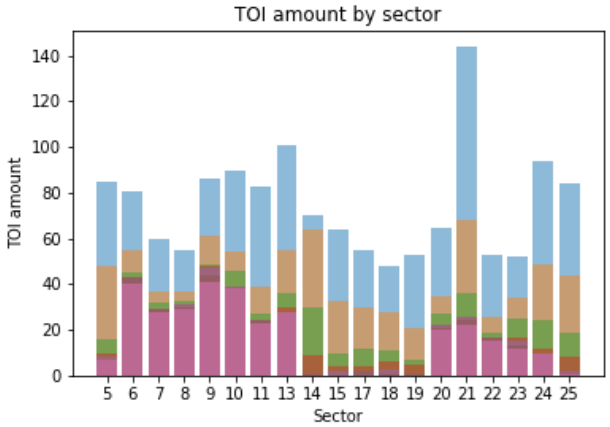


Figure 5: TOI-TCE match *amounts* by sector at 1% feature similarity

It is evident that, with a 10% threshold, many more matches are found. This is primarily because a large proportion of the data appears to be stored to few significant figures. Therefore, I chose to use a 10% threshold when labelling the final training set.

To account for variation between the sectors, I determined the best approximation for the correct labelling of TCEs would be through keeping the proportion of matches constant. For, although it is expected that the amount of positive cases should vary sector to sector, it is a reasonable approximation that the number of TOI-TCE matches remains constant relative to the number of total TCEs in a sector. This was done by selecting a relative ratio and choosing a different amount of features required for a match in different sectors. Alternative methods were tried, such as simply requiring a constant number of feature matches across sectors, however these training sets produced significantly worse final models.

I determined the appropriate proportion to be 5% and used Figure 6 to select the number of characteristics required for a match in each sector, the results of which can be found in Table 1. Figure 6 is similar to Figure 4 however, instead of plotting a raw amount, the proportion of matches is plotted instead.

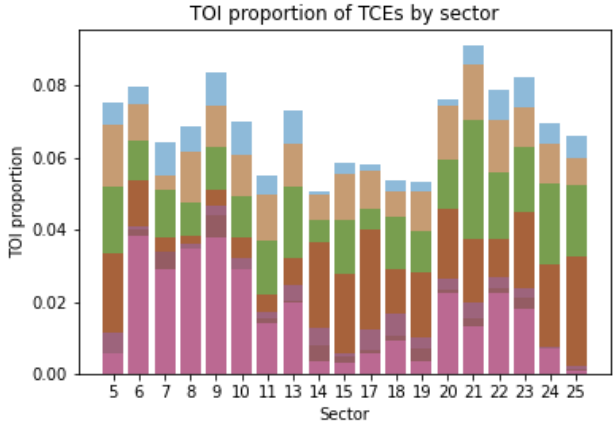


Figure 6: TOI-TCE match *proportions* by sector at 10% similarity

# Matches	Sectors
2	11, 14, 15, 17, 18 and 19
3	5, 7, 8, 10, 13, 20, 22, 24 and 25
4	6, 9, 21 and 23

Table 1: Sectors by number of feature matches required for a positive label

4.2 Neural Network Summary

I tested a variety of model architectures as I was concerned by the noise in some of the centroid measurements I looked at early on in the process. Therefore, I tested four architectures in total, which were various combinations of Light Curves (LCs), Centroids and Scientific Domain Parameters (SDPs). The variations were like so:

- Light Curve only
- Light Curve and Centroid Data
- Light Curve and SDPs
- Light Curve, Centroid Data and SDPs (All Data)

These differences in architecture produced surprising and significant differences in model performance and will be further explained in the coming sections. It is worth noting that all statistics in this section are derived from the aforementioned validation dataset.

4.2.1 Precision-Recall Curves

The performance of Neural Networks is often measured using a *precision-recall curve*. This is obtained by plotting the values found for precision against those of recall at different *thresholds* for determining whether a case was positive. Taking a threshold of 80% means that any outcome will be determined as positive if the model is above 80% certain of this classification, or negative otherwise. By taking many threshold values, we can calculate corresponding values for precision and recall, and plot them against one another to produce a curve. A rule of thumb for these curves is that the more they push towards the top right hand corner of the graph, the more reliable a model they represent. Precision and recall are used in this scenario because a measure of accuracy is not useful in this case. This is due to the large number of negative cases, meaning that a measure of accuracy does not inform us on how many of the few positive cases it is successfully identifying.

Precision is the probability of a correct evaluation, given that the evaluation is positive. Recall is the ratio of true positives to all the items the model evaluated to be positive. These can be written as the following equations:

$$\text{Precision} = \frac{\text{True Positives}}{\text{True Positives} + \text{False Positives}}$$

$$\text{Recall} = \frac{\text{True Positives}}{\text{True Positives} + \text{False Negatives}}$$

The results in Appendix D show the precision-recall curves for the eight models I trained of each of the four architectures. The thresholds used were simply each percentile from 0% to 99%. These plots show some useful information, such as that all the models following a single architecture behave similarly, yet they are still a little difficult to decipher. Following this, I plotted the median precision and recall value across all eight models for each threshold, allowing for a clearer comparison across different architectures. This can be seen in Figure 7. The simplest architecture performed better, which was somewhat unexpected. This says a lot about the way the data was processed and most likely the state of the centroids. This will be further examined in the interpretation section.

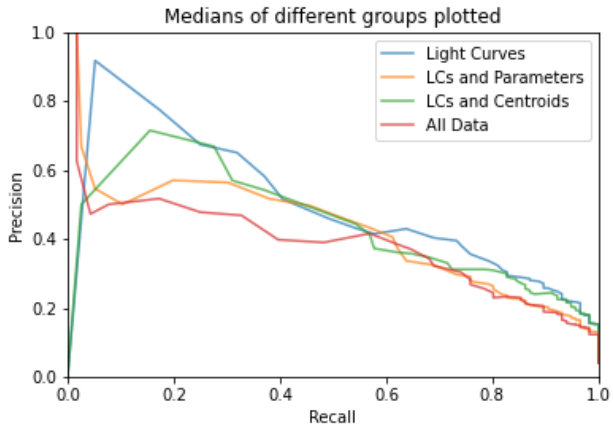


Figure 7: Median precision-recall curves of each architecture, with 0% to 99% threshold values. For all four architectures, precision increased as recall reduced, as would be expected - this trend is a positive sign and indicates that they can be improved upon.

It is also worth demonstrating the top end of performance for each architecture in greater detail, as there is variation between each percentile that cannot be identified in the first plot (Figure 7). Throughout these models, higher precision is found at higher threshold values. Therefore, Figure 8 is plotted using thresholds starting at 80%, going up to 100% in increments of 0.125%, to display the behaviour of models at higher precision more clearly. Figure 8 demonstrates that the simpler architectures, without the addition of parameters, consistently outperformed the other architectures at higher precision, rather than only across a couple of data points.

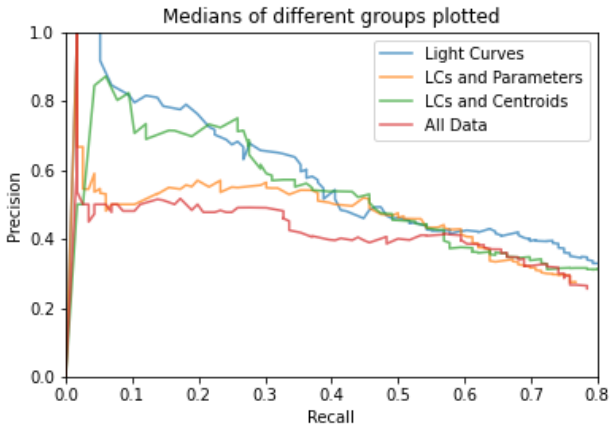


Figure 8: Median precision-recall curves of each architecture, with 80% to 100% threshold values. The parameter-based models can be seen to struggle to reach useful levels of precision even with low recall.

Finally, it is possible to condense the information contained in these graphs into a single score called *average-precision* (AP).

$$AP = \sum_n (R_n - R_{n-1}) P_n$$

Where R_n is recall at threshold n and P_n is the precision. The light curve only architecture attained a median AP of 53.14% and the highest performing model achieved an AP of 56.20%.

4.2.2 Confusion Matrices

Another common way to represent model performance is through a *confusion matrix*. This concisely sums up most classification problems and can really help to visualise those extending beyond a binary classification. Here, it is clear that all the models managed to learn something, as the vast majority of true negatives are marked negative.

Category

Threshold	Actual	Actual
Predicted	True Negative	False Negative
Predicted	False Positive	True Positive

LC only

85%	Neg.	Pos.
Neg.	1300.5	15.5
Pos.	60.5	42.5

LC only

95%	Neg.	Pos.
Neg.	1359.0	48.0
Pos.	2.0	10.0

LC + Centroids

85%	Neg.	Pos.
Neg.	1277.5	18.0
Pos.	83.5	40.0

LC + Centroids

95%	Neg.	Pos.
Neg.	1355.0	52.0
Pos.	6.0	6.0

LC + SDPs

85%	Neg.	Pos.
Neg.	1286.5	18.5
Pos.	74.5	39.5

LC + SDPs

95%	Neg.	Pos.
Neg.	1358.0	49.0
Pos.	3.0	9.0

LC, Centroids + SDPs

85%	Neg.	Pos.
Neg.	1277.0	18.0
Pos.	84.0	40.0

LC, Centroids + SDPs

95%	Neg.	Pos.
Neg.	1356.5	53.5
Pos.	4.5	4.5

Again, one can see that the distribution of True/False Positives/Negatives varies with the threshold for a prediction.

Through plotting the median of these values from all the models of one architecture against corresponding thresholds, we can observe how the different architectures behave. Interestingly, by examining behaviour at lower thresholds, we can see that there is some redemption for the models that include SDPs, as they are able to identify true negatives with greater certainty Figure 9. Alongside this, all models quickly identify the vast majority of true negative cases below the 20% threshold, whilst not producing any false negatives Figure 10. These results demonstrate the strength of these models for a triage process.

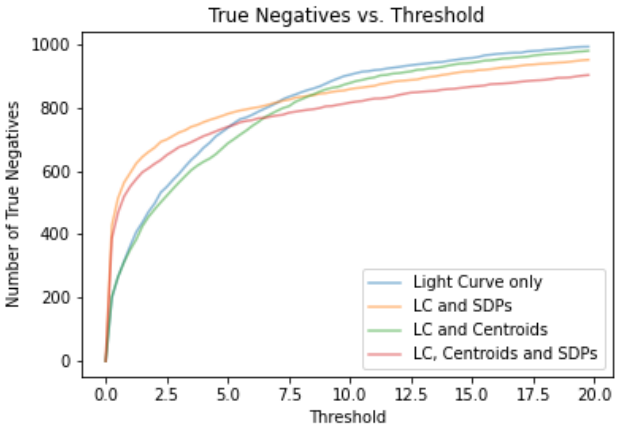


Figure 9: True negative rate at low thresholds – the parameter-based models are labelling negative cases more confidently, which is encouraging to see

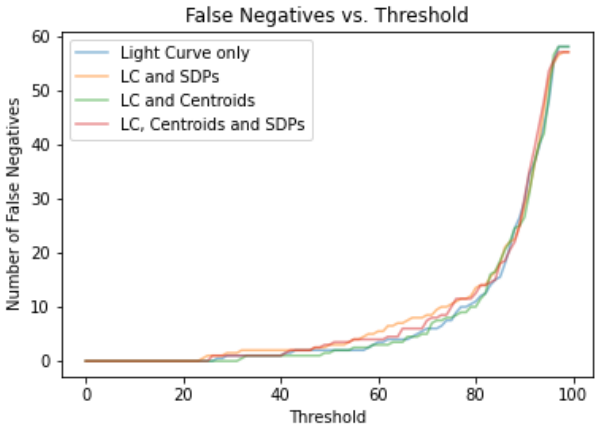


Figure 10: False negative rate at all thresholds – this demonstrates that the models only begin mislabelling above a threshold of 20%. In combination with Figure 9 showing that the majority of negative cases are predicted below 20%, this demonstrates the model's feasibility for triage

5 Interpretation

The main outcome of this project is that it has shown the feasibility of training a CNN with significantly less human vetting than previously thought. That is not to say that these models worked well, as clearly they did not perform sufficiently well to be useful for vetting. What it does show is that, with a mix of identification techniques, we can reduce the amount of human vetting required.

The root cause of the models' inaccuracies must be the training set and there are certainly some flaws in the process of its creation, which I will look to elaborate on in the coming section.

5.1 Training Set Evaluation

In general, I believe the creation of the training set to be relatively successful. What this report managed to demonstrate was the feasibility of automatic labelling of a training set, therefore reducing the time required to produce one. Due to the nature of this data, there are significantly fewer positive cases, therefore mislabelling by automatic processes can crucially influence the model. The fact this project was undertaken on a limited time scale meant that automation was the only solution, and this is likely the root cause of poor precision. Yet, in the future, I believe that augmenting human vetting process with the automatic method outlined in this report would be beneficial. By only vetting TCEs that have one feature match or more, the number that require vetting will reduce by an order of magnitude.

However, this was not the only problem with training set preparation as, due to the brevity of the project, some simplifying assumptions were made. As addressed earlier, I chose to only use the momentum-based centroids as opposed to the PSF centroids. Due to the data-downlink manoeuvre performed by TESS each orbit, the momentum measurements of centroids can have significant amounts of noise introduced. On top of this, TESS's centroid measurements are less precise than that of Kepler's due to the lower resolution of its cameras. I believe this noise is what led to the addition of centroids, having an overall negative effect on those models. A clear fix for the first problem would be to use the PSF centroids when available for a light curve, and momentum-based if not. Demonstrations of noisy centroids can be found in Appendix B.

There were also certainly ways to improve the time series processing. Firstly, the threshold requiring 50% of the data to be missing to discard a TCE may have been too low – missing even 5% could have a drastic effect on identification if it occurs across a transit window. Secondly, linearly interpolating across missing data is likely a suitable solution outside the transit window but, again, if within the transit window, it could have a significant effect on the shape of the transit. There are certainly alternative methods of interpolating missing data and perhaps some of these could be investigated in future work. Finally, the local view could perhaps have been too wide. For a view that is supposed to focus on the transit, the proportion that the transit took up was quite small. The transit would only be in an eighth of the bins, as the view was four transits wide either side of the center. Perhaps, in future work, it would make sense to change the local view to be more focused on the transit window to allow the CNN to examine the shape of the transit in more detail.

The Scientific Domain Parameters could have also been improved. These values were not always present where I expected to find them, leading to gaps in the dataset. These were filled in with a sentinel value of 0 as a solution in my models. If I was to use the median of the dataset, it would have simply been a value consistent with a negative case as these were the vast majority, hence I decided against doing so. Investigating an appropriate way to substitute missing values could be one future avenue of work.

Finally, it would be interesting to see if it is possible to reduce some of the data loss during the training set creation process. There was a steady pruning of data per sector for undetermined reasons. Even the first step to determine Observation IDs of TESS data products, where the search was most general and unrestricted, sometimes failed to turn up results. Fixing some of these data pruning issues would result in an expansion of the dataset and, as always with Machine Learning projects, more data would yield a more accurate model.

In conclusion, although this automatic method indicated some ways to reduce work to produce a training set, it is certainly not the final solution. Overall, it failed to yield the results desired and, without the addition of human vetting, I don't think it would be able to do so with other improvements either.

5.2 Neural Net Evaluation

In general, none of the models pass the bar of usefulness. However, we can see them managing to distinguish and confidently label a large portion of negative cases correctly, which is promising. With improvements to the dataset, I can see all these models yielding better results. On top of this, I think a lot of the variety we saw in model performance was more down to the underlying dataset, and I do not think it gave a true perspective on the performance of different architectures.

However, I believe this attempt has demonstrated that using a supervised CNN may be impractical for the PHT use case. This is primarily because a lot of the PHT data focuses on single transit observations. It is a difficult and inaccurate process to extract the information required to prepare these time series for consumption by a supervised CNN. As there are so many of these single transit observations, perhaps it could be worth attempting the use of an unsupervised CNN, using a training set of time series centered on the point of transit marked by PHT participants. Due to the volume of data produced, this seems like a viable option for an ML approach to this problem.

5.3 Comparison to Previous Attempts

Compared to previous attempts with Kepler data, this model was woefully inadequate as precision and recall were both very low. The reasons for this are specification differences between the two space telescopes outlined in subsection 1.3, primarily pixel size. However, compared to other attempts with real TESS data, the result of this project might not be entirely unexpected. Yu et al. (2019) demonstrated a similar result, as can be seen in Figure 11.

Although Yu et al. (2019) were working with fewer sectors (1 through 5), they had manually vetted their training set, which is likely why they saw better performance. However, the model was still largely on the same order of potential and this can be seen by the similarity of precision-recall curves in Figure 11. Comparing the most similar models between this report and Yu et al. (2019), we can see that the average precision is largely similar despite a lack of human vetting. The light curve only vetting model in this report achieving an average precision of 53.1% and the plain vetting model in Yu et al. (2019) achieving an average precision of 60.5%. However, Figure 11 shows that they had a very efficient triage model and that represents what a successful Neural Network should look like.

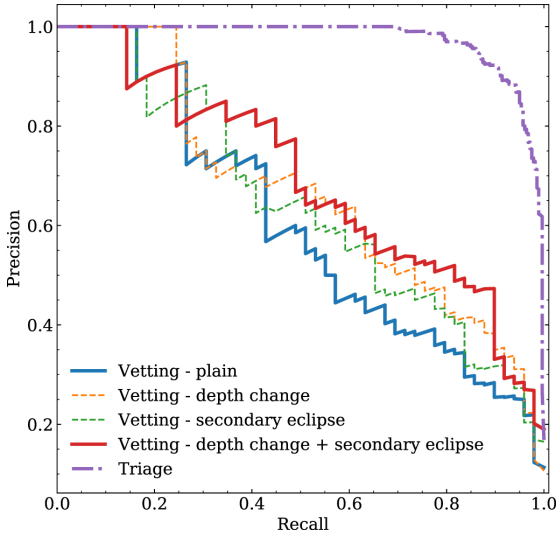


Figure 11: Precision-recall curve from Yu et al. (2019)

Alternatively in Osborn et al. (2020), they trained a model off purely simulated data. When applied to a sample of simulated data, they achieved promising results with an average precision of 97.3%. It is difficult to apply a model trained on simulated data to the real data, yet Osborn et al. (2020) managed to recover 61% of the TOI list existing at the time. Although this is an improvement upon Yu et al. (2019), it is still not usable for vetting as it is not fully reliable.

Finally, it's worth comparing to the work of the previous MPhys candidate (Anon. CN: 1015761, 2020). They used two methods, Random Forest (RF) and Probability Random Forest (PRF). The RF method achieved better results, giving a precision of 97.2% and a recall of 96.8%. Evidently, this achieved brilliant results for vetting, however it was too difficult to apply to the PHT use case.

6 Conclusion

In conclusion, despite the project failing to yield a successful model, it did provide insight into the automation of training set creation. It is also clear that there are many paths to improving upon this model, if that path is chosen. However, I believe that the literature and this report indicates that the TESS data (in particular, the centroid information) is too noisy to train a successful model for vetting. They may however be useful in triage.

As PHT requires a vetting model, this may indicate a supervised approach is not the correct one. Were an unsupervised attempt to be made, it would allow the use of the full set of light curves and not just the TCEs, producing a notably larger training set. Therefore, I think it could be worth investigating an unsupervised deep learning approach, as it might be more appropriate for the PHT use case.

7 Acknowledgements

None of this work would have been possible without the consistent support offered by my supervisors Suzanne Aigrain and Oscar Barragán. Many thanks to them for giving me this opportunity to work alongside them and in researching an incredibly interesting topic.

References

Anon. CN: 1015761 (2020). Detection of transiting exoplanets with the tess space mission. `https://github.com/1015761/MPhys/blob/master/Project.pdf`.

Anon. CN: 1025032 (2021). Github repository accompanying this report. `https://github.com/10108/MPhys-Project`.

Ansdell, M., Ioannou, Y., Osborn, H. P., Sasdelli, M., 2018 NASA Frontier Development Lab Exoplanet Team, Smith, J. C., Caldwell, D., Jenkins, J. M., Räissi, C., Angerhausen, D., and NASA Frontier Development Lab Exoplanet Mentors, . (2018). Scientific Domain Knowledge Improves Exoplanet Transit Classification with Deep Learning. *ApJL*, 869(1):L7.

Auvergne, M., Bodin, P., Boisnard, L., Buey, J. T., Chaintreuil, S., Epstein, G., Jouret, M., Lam-Trong, T., Levacher, P., Magnan, A., Perez, R., Plasson, P., Plesseria, J., Peter, G., Steller, M., Tiphène, D., Baglin, A., Agogué, P., Appourchaux, T., Barbet, D., Beaufort, T., Bellenger, R., Berlin, R., Bernardi, P., Blouin, D., Boumier, P., Bonneau, F., Briet, R., Butler, B., Cautain, R., Chiavassa, F., Costes, V., Cuvilho, J., Cunha-Parro, V., de Oliveira Fialho, F., Decaudin, M., Defise, J. M., Djalal, S., Docclo, A., Drummond, R., Dupuis, O., Exil, G., Fauré, C., Gaboriaud, A., Gamet, P., Gavalda, P., Grolleau, E., Gueguen, L., Guivarc'h, V., Guterman, P., Hasiba, J., Huntzinger, G., Hustaix, H., Imbert, C., Jeanville, G., Johlander, B., Jorda, L., Journoud, P., Karioty, F., Kerjean, L., Lafond, L., Lapeyrere, V., Landiech, P., Larqué, T., Laudet, P., Le Merrer, J., Leporati, L., Leruyet, B., Levieuge, B., Llebaria, A., Martin, L., Mazy, E., Mesnager, J. M., Michel, J. P., Moalic, J. P., Monjoin, W., Naudet, D., Neukirchner, S., Nguyen-Kim, K., Ollivier, M., Orcesi, J. L., Ottacher, H., Oulali, A., Parisot, J., Perruchot, S., Piacentino, A., Pinheiro da Silva, L., Platzer, J., Pontet, B., Pradines, A., Quentin, C., Rohbeck, U., Rolland, G., Rollenhagen, F., Romagnan, R., Russ, N., Samadi, R., Schmidt, R., Schwartz, N., Sebbag, I., Smit, H., Sunter, W., Tello, M., Toulouse, P., Ulmer, B., Vandermarcq, O., Vergnault, E., Wallner, R., Waultier, G., and Zanatta, P. (2009). The CoRoT satellite in flight: description and performance. *A&A*, 506(1):411–424.

Catanzarite, J. H. (2015). Autovetter planet candidate catalog for q1-q17 data release 24 (ksci-19090-001). `https://exoplanetarchive.ipac.caltech.edu/docs/KSCI-19091-001.pdf`.

Chauvin, G., Lagrange, A. M., Dumas, C., Zuckerman, B., Mouillet, D., Song, I., Beuzit, J. L., and Lowrance, P. (2004). A giant planet candidate near a young brown dwarf. Direct VLT/NACO observations using IR wavefront sensing. *A&A*, 425:L29–L32.

Dattilo, A., Vanderburg, A., Shallue, C. J., Mayo, A. W., Berlind, P., Bieryla, A., Calkins, M. L., Esquerdo, G. A., Everett, M. E., Howell, S. B., Latham, D. W., Scott, N. J., and Yu, L. (2019). Identifying Exoplanets with Deep Learning. II. Two New Super-Earths Uncovered by a Neural Network in K2 Data. *AJ*, 157(5):169.

Eisner, N. L., Barragán, O., Aigrain, S., Lintott, C., Miller, G., Zicher, N., Boyajian, T. S., Briceño, C., Bryant, E. M., Christiansen, J. L., Feinstein, A. D., Flor-Torres, L. M., Fridlund, M., Gandolfi, D., Gilbert, J., Guerrero, N., Jenkins, J. M., Jones, K., Kristiansen, M. H., Vanderburg, A., Law, N., López-Sánchez, A. R., Mann, A. W., Safron, E. J., Schwamb, M. E., Stassun, K. G., Osborn, H. P., Wang, J., Zic, A., Ziegler, C., Barnet, F., Bean, S. J., Bundy, D. M., Chetnik, Z., Dawson, J. L., Garstone, J., Stenner, A. G., Huten, M., Larish, S., Melanson, L. D., Mitchell, T., Moore, C., Peltsch, K., Rogers, D. J., Schuster, C., Smith, D. S., Simister, D. J., Tanner, C., Terentev, I., and Tsymbal, A. (2020). Planet Hunters TESS I: TOI 813, a subgiant hosting a transiting Saturn-sized planet on an 84-day orbit. *MNRAS*, 494(1):750–763.

Eisner, N. L., Barragán, O., Lintott, C., Aigrain, S., Nicholson, B., Boyajian, T. S., Howell, S., Johnston, C., Lakeland, B., Miller, G., McMaster, A., Parviainen, H., Safron, E. J., Schwamb, M. E., Trouille, L., Vaughan, S., Zicher, N., Allen, C., Allen, S., Bouslog, M., Johnson, C., Simon, M. N., Wolfenbarger, Z., Baeten, E. M. L., Bundy, D. M., and Hoffman, T. (2021). Planet Hunters TESS II: findings from the first two years of TESS. *MNRAS*, 501(4):4669–4690.

Guerrero, N. M., Seager, S., Huang, C. X., Vanderburg, A., Garcia Soto, A., Mireles, I., Hesse, K., Fong, W., Glidden, A., Shporer, A., Latham, D. W., Collins, K. A., Quinn, S. N., Burt, J., Dragomir, D., Crossfield, I., Vanderspek, R., Fausnaugh, M., Burke, C. J., Ricker, G., Daylan, T., Essack, Z., Günther, M. N., Osborn, H. P., Pepper, J., Rowden, P., Sha, L., Villanueva, Steven, J., Yahalomi, D. A., Yu, L., Ballard, S., Batalha, N. M., Berardo, D., Chontos, A., Dittmann, J. A., Esquerdo, G. A., Mikal-Evans, T., Jayaraman, R., Krishnamurthy, A., Louie, D. R., Mehrle, N., Niraula, P., Rackham, B. V., Rodriguez, J. E., Rowden, S. J. L., Sousa-Silva, C., Watanabe, D., Wong, I., Zhan, Z., Zivanovic, G., Christiansen, J. L., Ciardi, D. R., Swain, M. A., Lund, M. B., Mullally, S. E., Fleming, S. W., Rodriguez, D. R., Boyd, P. T., Quintana, E. V., Barclay, T., Colón, K. D., Rinehart, S. A., Schlieder, J. E., Clampin, M., Jenkins, J. M., Twicken, J. D., Caldwell, D. A., Coughlin, J. L., Henze, C., Lissauer, J. J., Morris, R. L., Rose, M. E., Smith, J. C., Tenenbaum, P., Ting, E. B., Wohler, B., Bakos, G. Á., Bean, J. L., Berta-Thompson, Z. K., Bieryla, A., Bouma, L. G., Buchhave, L. A., Butler, N., Charbonneau, D., Doty, J. P., Ge, J., Holman, M. J., Howard, A. W., Kaltenegger, L., Kane, S. R., Kjeldsen, H., Kreidberg, L., Lin, D. N. C., Minsky, C., Narita, N., Paegert, M., Pál, A., Palle, E., Sasselov, D. D., Spencer, A., Sozzetti, A., Stassun, K. G., Torres, G., Udry, S., and Winn, J. N. (2021). The TESS Objects of Interest Catalog from the TESS Prime Mission. *arXiv e-prints*, page arXiv:2103.12538.

Han, J. and Moraga, C. (1995). The influence of the sigmoid function parameters on the speed of backpropagation learning. In Mira, J. and Sandoval, F., editors, *From Natural to Artificial Neural Computation*, pages 195–201, Berlin, Heidelberg. Springer Berlin Heidelberg.

Howell, S. B., Sobeck, C., Haas, M., Still, M., Barclay, T., Mullally, F., Troeltzsch, J., Aigrain, S., Bryson, S. T., Caldwell, D., Chaplin, W. J., Cochran, W. D., Huber, D., Marcy, G. W., Miglio, A., Najita, J. R., Smith, M., Twicken, J. D., and Fortney, J. J. (2014). The K2 Mission: Characterization and Early Results. *PASP*, 126(938):398.

Jenkins, J. M., Caldwell, D. A., Chandrasekaran, H., Twicken, J. D., Bryson, S. T., Quintana, E. V., Clarke, B. D., Li, J., Allen, C., Tenenbaum, P., Wu, H., Klaus, T. C., Middour, C. K., Cote, M. T., McCauliff, S., Girouard, F. R., Gunter, J. P., Wohler, B., Sommers, J., Hall, J. R., Uddin, A. K., Wu, M. S., Bhavsar, P. A., Van Cleve, J., Pletcher, D. L., Dotson, J. A., Haas, M. R., Gilliland, R. L., Koch, D. G., and Borucki, W. J. (2010). Overview of the Kepler Science Processing Pipeline. *ApJL*, 713(2):L87–L91.

Jenkins, J. M., Twicken, J. D., McCauliff, S., Campbell, J., Sanderfer, D., Lung, D., Mansouri-Samani, M., Girouard, F., Tenenbaum, P., Klaus, T., Smith, J. C., Caldwell, D. A., Chacon, A. D., Henze, C., Heiges, C., Latham, D. W., Morgan, E., Swade, D., Rinehart, S., and Vanderspek, R. (2016). The TESS science processing operations center. In Chiozzi, G. and Guzman, J. C., editors, *Software and Cyberinfrastructure for Astronomy IV*, volume 9913 of *Society of Photo-Optical Instrumentation Engineers (SPIE) Conference Series*, page 99133E.

Kingma, D. P. and Ba, J. (2014). Adam: A Method for Stochastic Optimization. *arXiv e-prints*, page arXiv:1412.6980.

Koch, D. G., Borucki, W., Dunham, E., Geary, J., Gilliland, R., Jenkins, J., Latham, D., Bachtell, E., Berry, D., Deininger, W., Duren, R., Gautier, T. N., Gillis, L., Mayer, D., Miller, C. D., Shafer, D., Sobeck, C. K., Stewart, C., and Weiss, M. (2004). Overview and status of the Kepler Mission. In Mather, J. C., editor, *Optical, Infrared, and Millimeter Space Telescopes*, volume 5487 of *Society of Photo-Optical Instrumentation Engineers (SPIE) Conference Series*, pages 1491–1500.

Konacki, M., Torres, G., Jha, S., and Sasselov, D. D. (2003). An extrasolar planet that transits the disk of its parent star. *Natur*, 421(6922):507–509.

Liu, L. and Qi, H. (2017). Learning effective binary descriptors via cross entropy. In *2017 IEEE Winter Conference on Applications of Computer Vision (WACV)*, pages 1251–1258.

MAST (2020). Mast tce bulk downloads. `https://archive.stsci.edu/tess/bulk_downloads/bulk_downloads_tce.html`. [Online; accessed multiple dates].

Mayor, M. and Queloz, D. (1995). A Jupiter-mass companion to a solar-type star. *Natur*, 378(6555):355–359.

McCauliff, S. D., Jenkins, J. M., Catanzarite, J., Burke, C. J., Coughlin, J. L., Twicken, J. D., Tenenbaum, P., Seader, S., Li, J., and Cote, M. (2015). Automatic Classification of Kepler Planetary Transit Candidates. *ApJ*, 806(1):6.

Murray, C. D. and Correia, A. C. M. (2010). *Keplerian Orbits and Dynamics of Exoplanets*, pages 15–23.

Osborn, H. P., Ansdell, M., Ioannou, Y., Sasdelli, M., Angerhausen, D., Caldwell, D., Jenkins, J. M., Räissi, C., and Smith, J. C. (2020). Rapid classification of TESS planet candidates with convolutional neural networks. *A&A*, 633:A53.

Pepe, F. A., Cristiani, S., Rebolo Lopez, R., Santos, N. C., Amorim, A., Avila, G., Benz, W., Bonifacio, P., Cabral, A., Carvas, P., Cirami, R., Coelho, J., Comari, M., Coretti, I., De Caprio, V., Dekker, H., Delabre, B., Di Marcantonio, P., D'Odorico, V., Fleury, M., García, R., Herreros Linares, J. M., Hughes, I., Iwert, O., Lima, J., Lizon, J.-L., Lo Curto, G., Lovis, C., Manescau, A., Martins, C., Mégevand, D., Moitinho, A., Molaro, P., Monteiro, M., Monteiro, M., Pasquini, L., Mordasini, C., Queloz, D., Rasilla, J. L., Rebordão, J. M., Santana Tschudi, S., Santin, P., Sosnowska, D., Spanò, P., Tenegi, F., Udry, S., Vanzella, E., Viel, M., Zapatero Osorio, M. R., and Zerbi, F. (2010). ESPRESSO: the Echelle spectrograph for rocky exoplanets and stable spectroscopic observations. In McLean, I. S., Ramsay, S. K., and Takami, H., editors, *Ground-based and Airborne Instrumentation for Astronomy III*, volume 7735 of *Society of Photo-Optical Instrumentation Engineers (SPIE) Conference Series*, page 77350F.

Ramachandran, P., Zoph, B., and Le, Q. V. (2017). Searching for Activation Functions. *arXiv e-prints*, page arXiv:1710.05941.

Rauer, H., Catala, C., Aerts, C., Appourchaux, T., Benz, W., Brandeker, A., Christensen-Dalsgaard, J., Deleuil, M., Gizon, L., Goupil, M. J., Güdel, M., Janot-Pacheco, E., Mas-Hesse, M., Pagano, I., Piotto, G., Pollacco, D., Santos, Ċ., Smith, A., Suárez, J. C., Szabó, R., Udry, S., Adibekyan, V., Alibert, Y., Almenara, J. M., Amaro-Seoane, P., Eiff, M. A.-v., Asplund, M., Antonello, E., Barnes, S., Baudin, F., Belkacem, K., Bergemann, M., Bihain, G., Birch, A. C., Bonfils, X., Boisse, I., Bonomo, A. S., Borsa, F., Brandão, I. M., Brocato, E., Brun, S., Burleigh, M., Burston, R., Cabrera, J., Cassisi, S., Chaplin, W., Charpinet, S., Chiappini, C., Church, R. P., Csizmadia, S., Cunha, M., Damasso, M., Davies, M. B., Deeg, H. J., Díaz, R. F., Dreizler, S., Dreyer, C., Eggenberger, P., Ehrenreich, D., Eigmüller, P., Erikson, A., Farmer, R., Feltzing, S., de Oliveira Fialho, F., Figueira, P., Forveille, T., Fridlund, M.,

García, R. A., Giommi, P., Giuffrida, G., Godolt, M., Gomes da Silva, J., Granzer, T., Grenfell, J. L., Grotsch-Noels, A., Günther, E., Haswell, C. A., Hatzes, A. P., Hébrard, G., Hekker, S., Helled, R., Heng, K., Jenkins, J. M., Johansen, A., Khodachenko, M. L., Kislyakova, K. G., Kley, W., Kolb, U., Krivova, N., Kupka, F., Lammer, H., Lanza, A. F., Lebreton, Y., Magrin, D., Marcos-Arenal, P., Marrese, P. M., Marques, J. P., Martins, J., Mathis, S., Mathur, S., Messina, S., Miglio, A., Montalban, J., Montalto, M., Monteiro, M. J. P. F. G., Moradi, H., Moravveji, E., Mordasini, C., Morel, T., Mortier, A., Nascimbeni, V., Nelson, R. P., Nielsen, M. B., Noack, L., Norton, A. J., Ofir, A., Oshagh, M., Ouazzani, R. M., Pápics, P., Parro, V. C., Petit, P., Plez, B., Poretti, E., Quirrenbach, A., Ragazzoni, R., Raimondo, G., Rainer, M., Reese, D. R., Redmer, R., Reffert, S., Rojas-Ayala, B., Roxburgh, I. W., Salmon, S., Santerne, A., Schneider, J., Schou, J., Schuh, S., Schunker, H., Silva-Valio, A., Silvotti, R., Skillen, I., Snellen, I., Sohl, F., Sousa, S. G., Sozzetti, A., Stello, D., Strassmeier, K. G., Švanda, M., Szabó, G. M., Tkachenko, A., Valencia, D., Van Grootel, V., Vauclair, S. D., Ventura, P., Wagner, F. W., Walton, N. A., Weingrill, J., Werner, S. C., Wheatley, P. J., and Zwintz, K. (2014). The PLATO 2.0 mission. *Experimental Astronomy*, 38(1-2):249–330.

Reis, I., Baron, D., and Shahaf, S. (2019). Probabilistic Random Forest: A Machine Learning Algorithm for Noisy Data Sets. *AJ*, 157(1):16.

Ricker, G. R., Winn, J. N., Vanderspek, R., Latham, D. W., Bakos, G. Á., Bean, J. L., Berta-Thompson, Z. K., Brown, T. M., Buchhave, L., Butler, N. R., Butler, R. P., Chaplin, W. J., Charbonneau, D., Christensen-Dalsgaard, J., Clampin, M., Deming, D., Doty, J., De Lee, N., Dressing, C., Dunham, E. W., Endl, M., Fressin, F., Ge, J., Henning, T., Holman, M. J., Howard, A. W., Ida, S., Jenkins, J., Jernigan, G., Johnson, J. A., Kaltenegger, L., Kawai, N., Kjeldsen, H., Laughlin, G., Levine, A. M., Lin, D., Lissauer, J. J., MacQueen, P., Marcy, G., McCullough, P. R., Morton, T. D., Narita, N., Paegert, M., Palle, E., Pepe, F., Pepper, J., Quirrenbach, A., Rinehart, S. A., Sasselov, D., Sato, B., Seager, S., Sozzetti, A., Stassun, K. G., Sullivan, P., Szentgyorgyi, A., Torres, G., Udry, S., and Villasenor, J. (2014). Transiting Exoplanet Survey Satellite (TESS). In Oschmann, Jacobus M., J., Clampin, M., Fazio, G. G., and MacEwen, H. A., editors, *Space Telescopes and Instrumentation 2014: Optical, Infrared, and Millimeter Wave*, volume 9143 of *Society of Photo-Optical Instrumentation Engineers (SPIE) Conference Series*, page 914320.

Shallue, C. J. and Vanderburg, A. (2018). Identifying Exoplanets with Deep Learning: A Five-planet Resonant Chain around Kepler-80 and an Eighth Planet around Kepler-90. *AJ*, 155(2):94.

Stassun, K. G., Oelkers, R. J., Pepper, J., Paegert, M., De Lee, N., Torres, G., Latham, D. W., Charpinet, S., Dressing, C. D., Huber, D., Kane, S. R., Lépine, S., Mann, A., Muirhead, P. S., Rojas-Ayala, B., Silvotti, R., Fleming, S. W., Levine, A., and Plavchan, P. (2018). The TESS Input Catalog and Candidate Target List. *AJ*, 156(3):102.

Winn, J. N. (2010). Transits and Occultations. *arXiv e-prints*, page arXiv:1001.2010.

Yu, L., Vanderburg, A., Huang, C., Shallue, C. J., Crossfield, I. J. M., Gaudi, B. S., Daylan, T., Dattilo, A., Armstrong, D. J., Ricker, G. R., Vanderspek, R. K., Latham, D. W., Seager, S., Dittmann, J., Doty, J. P., Glidden, A., and Quinn, S. N. (2019). Identifying Exoplanets with Deep Learning. III. Automated Triage and Vetting of TESS Candidates. *AJ*, 158(1):25.

Appendices

A Software Usage

Code used in this project can be found in the GitHub repository located at Anon. CN: 1025032 (2021), this project was solely written in Python. The machine learning section grew from Exonet made publicly available by Ansdell et al. (2018). The code I wrote relied on several Python libraries and links to their Python package index (PyPI) can be found here:

- Astropy 4.2 - `https://pypi.org/project/astropy/`
- Astroquery 0.4.1 - `https://pypi.org/project/astroquery/`
- Matplotlib 3.3.3 - `https://pypi.org/project/matplotlib/`
- Numpy 1.19.2 - `https://pypi.org/project/numpy/`
- Pandas 1.1.5 - `https://pypi.org/project/pandas/`
- Scipy 1.5.4 - `https://pypi.org/project/scipy/`
- Sklearn (deprecated, used very little) - `https://pypi.org/project/sklearn/`
- Torch 1.7.1 - `https://pypi.org/project/torch/`
- tqdm 4.55.0 - `https://pypi.org/project/tqdm/`

B Time Series Examples

In this appendix one can find a number of examples for Light Curves and Centroid Time Series produced by various classifications. Typically the difference in shape for AFP and PC candidates is more noticeable in the local view. **TIC 104024556** - Good examples of how both processed light curves and centroids should look for a PC.

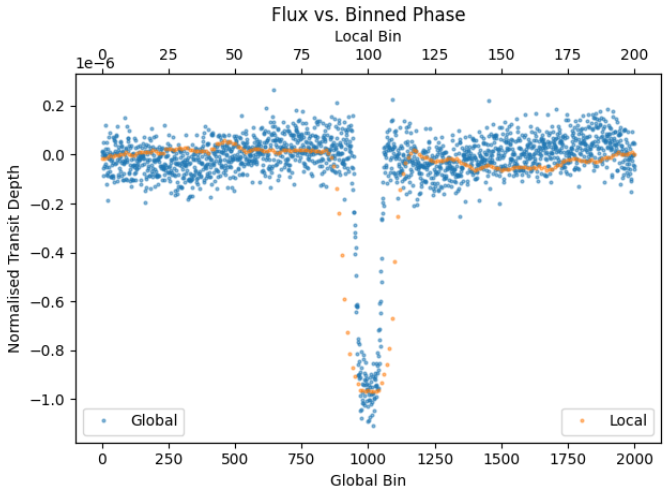


Figure 12: TIC 104024556 - Processed PC light flux

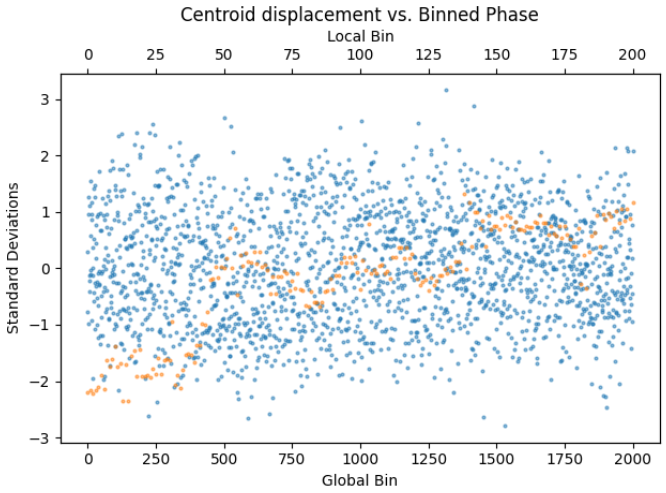


Figure 13: TIC 104024556 - Processed PC centroid

TIC 57151429 - Random noise that has been picked up as a TCE

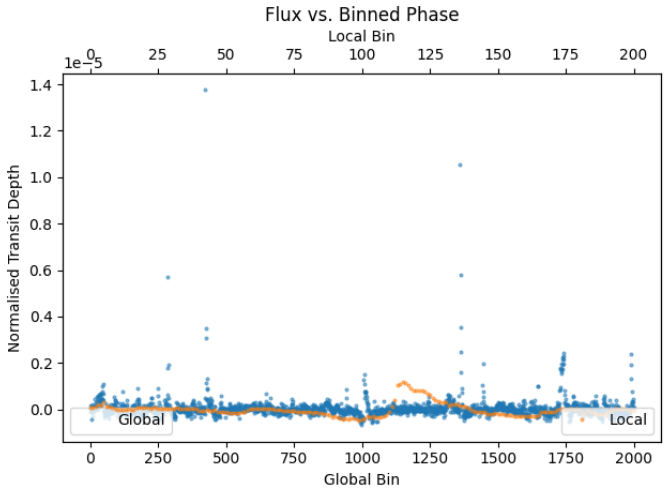


Figure 14: TIC 57151429 - Processed NTP light flux, visible excitation around center is the supposed transit

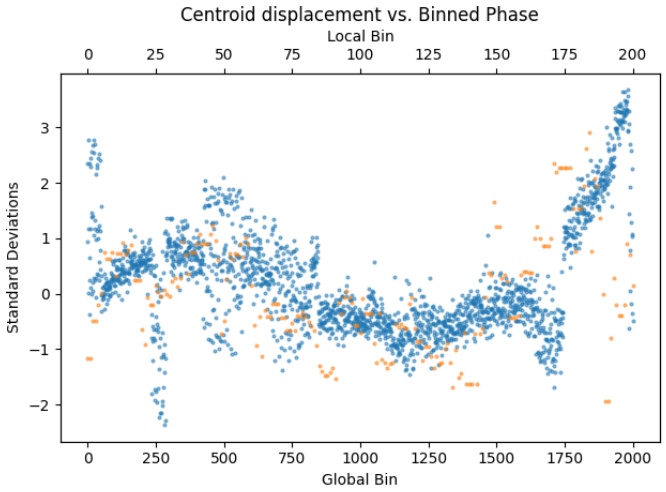


Figure 15: TIC 57151429 - Processed NTP centroid

TIC 438942374 - Background eclipsing binary displaying a signature sharp dip in flux time series, more noticeable in the local view.

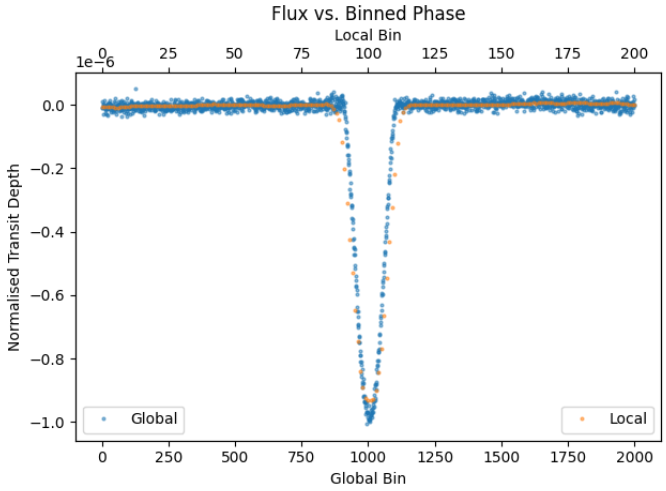


Figure 16: TIC 438942374 - Processed AFP light flux, sharp transit shape visible.

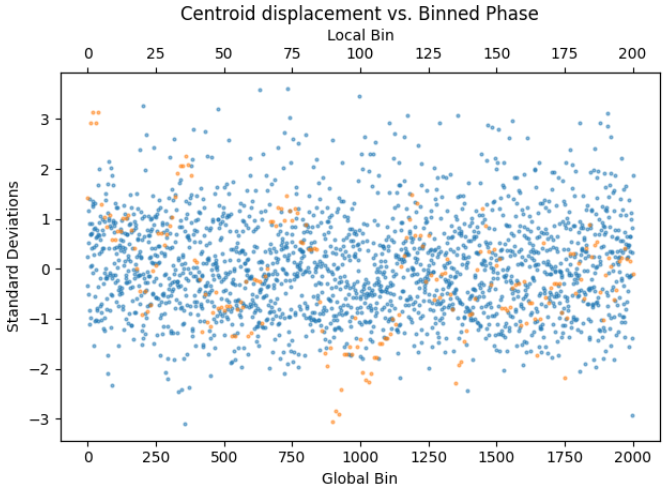


Figure 17: TIC 438942374 - Processed AFP centroid

TIC 304042899 - PC with difficult to discern centroid measurement

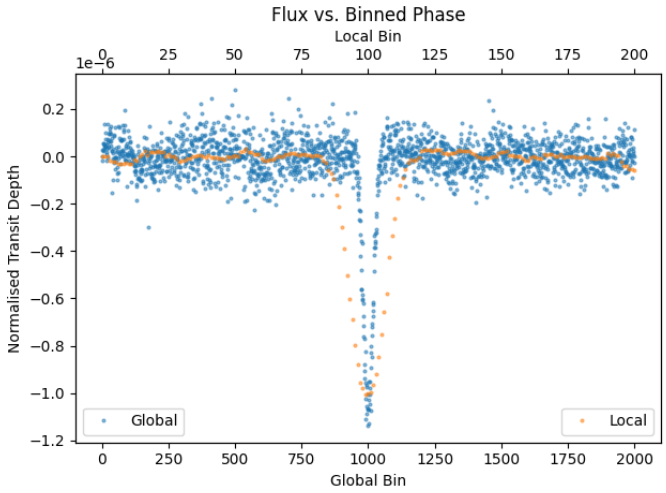


Figure 18: TIC 304042899 - Processed PC light flux

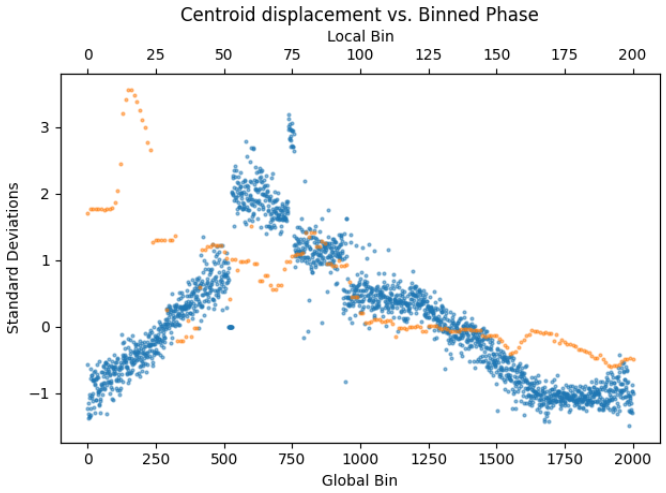
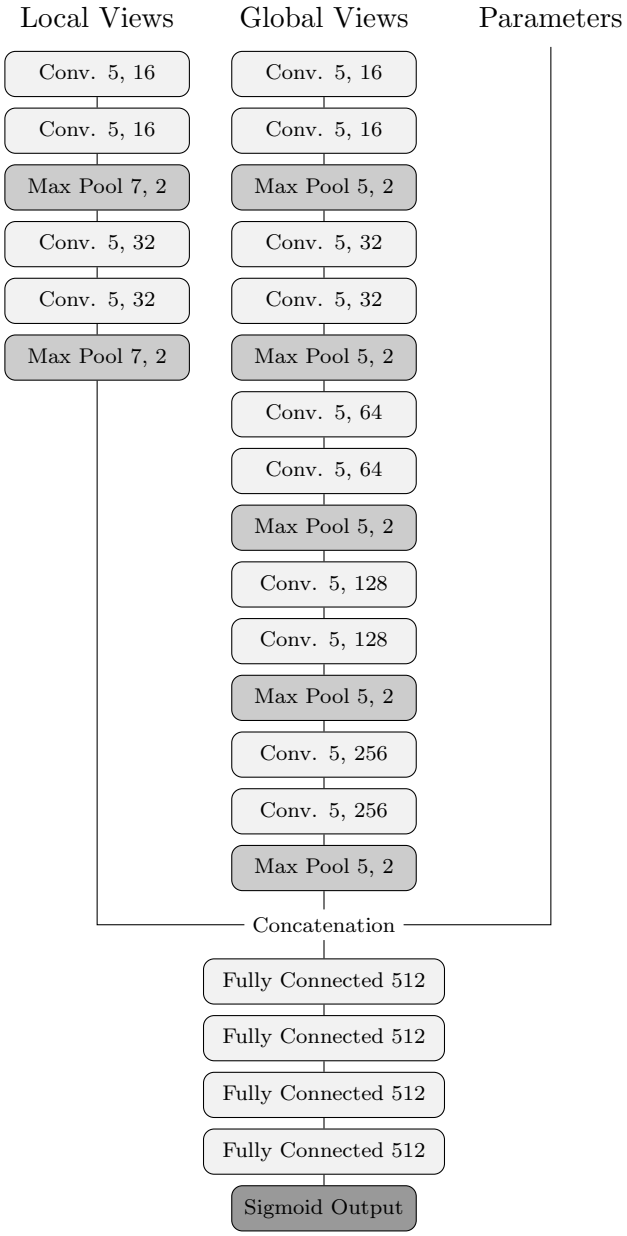


Figure 19: TIC 304042899 - Processed PC centroid, very noisy measurement

C Model Architecture

This is the architecture of the models used. Some models did not include the extra parameters, some models did not include centroid data in the local and global views.



D Precision-Recall Curves

Precision-recall curves for all models produced can be found in Figure 20. One can see a variety of performance even within a single architecture. There are clear similarities in the shapes of the curves for the two centroid based models (LCs and Centroids, and All Data), indicating their was significant difference between architectures beyond random noise.

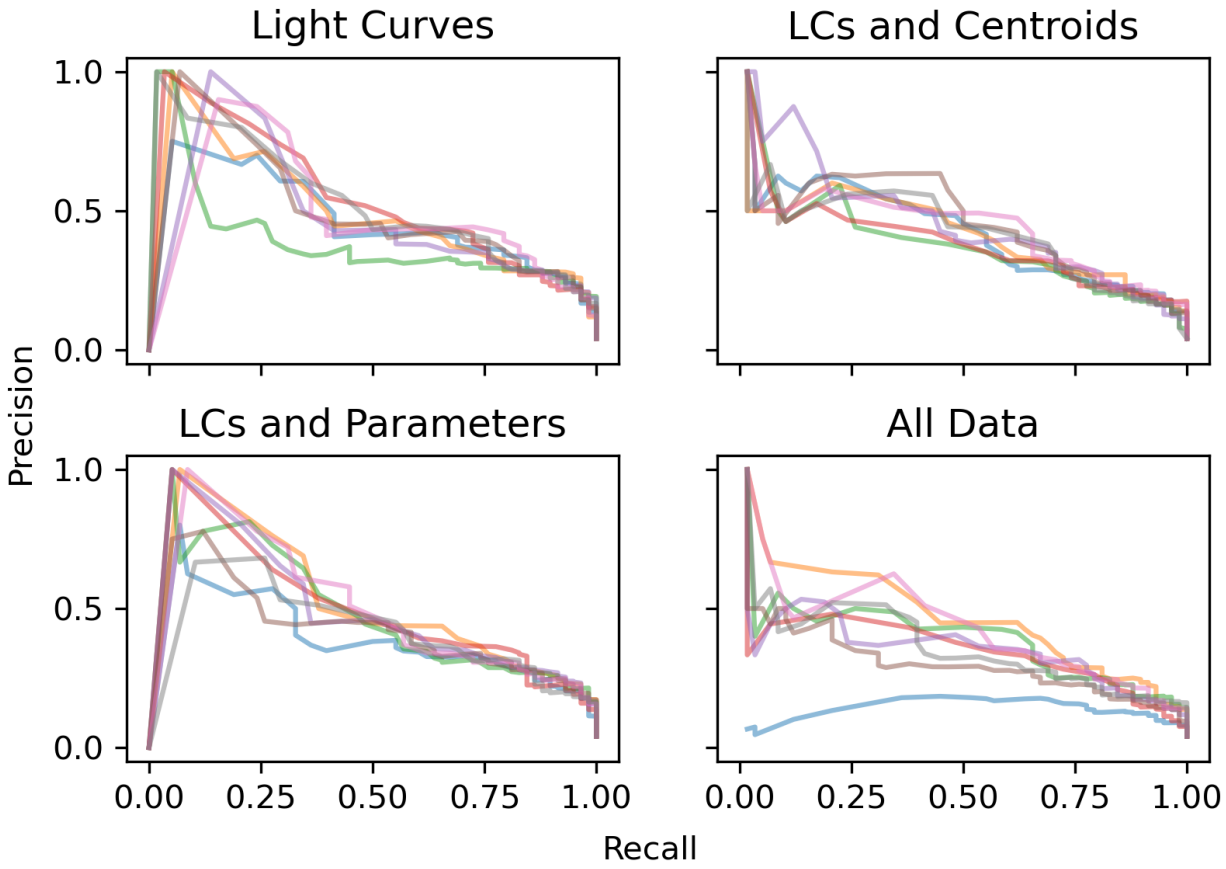


Figure 20: Precision recall Curves