Final Report on Astronomical Classification

Group #47:
Alexander Demidov[1], Wenqin Hu[2], Yang Zeng[3], Mier Chen[4]

1 Introduction

In this project, we create deep learning models for classifying astronomical objects based on sequences of images of these objects that would, in practice, be captured by a telescope. We use simulated image sequences generated with a process that matches real observation conditions. Using simulated data has two major benefits: it provides researchers with many examples to train a classifier, and it can be created with the camera and exposure parameters for telescopes that will come online in the future and have not yet collected any real data.

Classification of astronomical events by using sequences of images directly as inputs was first reported in 2018 by Carrasco-David et.al [1]. Another major contribution of this paper was the development of a method for generating simulated data sets of image sequences, as mentioned above. Traditional methods to classify astronomical objects are based on generating a time series called light curve as described in Naylor 1998 [2] and extracting features from the light curve that can be inputted into a statistical learning method, such as a random forest. Deep learning have also been tried for light curve classification [3].

Our two major goals in the project were: 1) to compare the performance of a deep learning model trained on raw images with the performance of a model trained on pre-preprocessed images, and 2) to investigate the effects of varying the temporal observation sequences that a model receives as inputs during training and testing. For our model architecture we used recurrent convolutional neural networks (RCNN) that have been successfully applied to learn time dependencies in other domains, such as action recognition in videos [4][5] and speech recognition [6].

We found that image pre-processing and the use of a randomized set of temporal observation sequences during training helped our RCNN model to achieve better accuracy when classifying test set sequences with time intervals different from those seen in training. These consideration are important to take into account when training a model on synthetic data that would later be employed to make prediction on real data. If we consider data that will

[1] Alexander_tkd@yahoo.com

[2] victorzeng811@gmail.com

[3] huwenqin74@gmail.com

[4] ccihiocil1@gmail.com

be collected in the future, it is likely that actual astronomical observations will come with temporal intervals between observations that are not known beforehand. In such situations it would be advantageous to have a model that is not overfit to one particular set of time intervals during training.

2 Exploratory Data Analysis

Object Class	**Description**
EmptyLigh	a control class with sequences of some random light patterns
Asteroids	an asteroid that appears in just one out of the 48 observation days
Constant	a non-variable star (this class may also include non-variable galaxies)
M33Cephei	a Cepheid variable in the M33 galaxy, a variable star that pulsates periodically (due to expansion and contraction of surface layers) in a predictable way; typical period: 2 to 60 days, average luminosity: 300 to 40,000 suns
RRLyrae	an RR Lyrae variable, a variable star of another type, also pulsates periodically; typical period: 4 hours to 1 day, average luminosity: 80suns
Supernova	a star that suddenly increases greatly in brightness because of a catastrophic explosion that ejects most of its mass; for a supernova event luminosity quickly builds up to a peak and then drops off gradually

Table 1 *Description of astronomical object classes in the data set*

Our data set contains 72,000 simulated image sequences of astronomical events. There are six object classes as described in Table 1 with 12,000 examples of each in the data set. Hence, the distribution of classes is balanced. The images were synthesized by taking known celestial objects time variability, real observing conditions and observing sampling and constructing realistic sky images at different times. Each sequence contains 48 images of 21x21 pixels.

A sample sequence is shown in Figure 1.

Observations were conducted on dates shown in Figure 2 with unequal time intervals between observation instances.

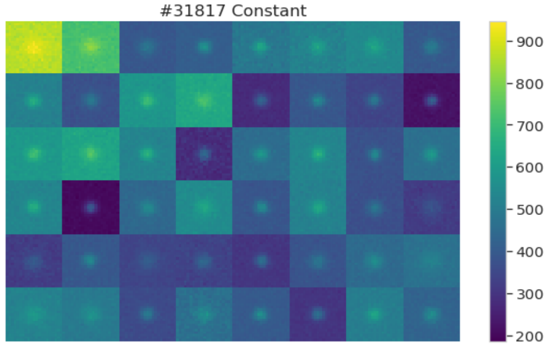

Figure 1 Sample Sequence

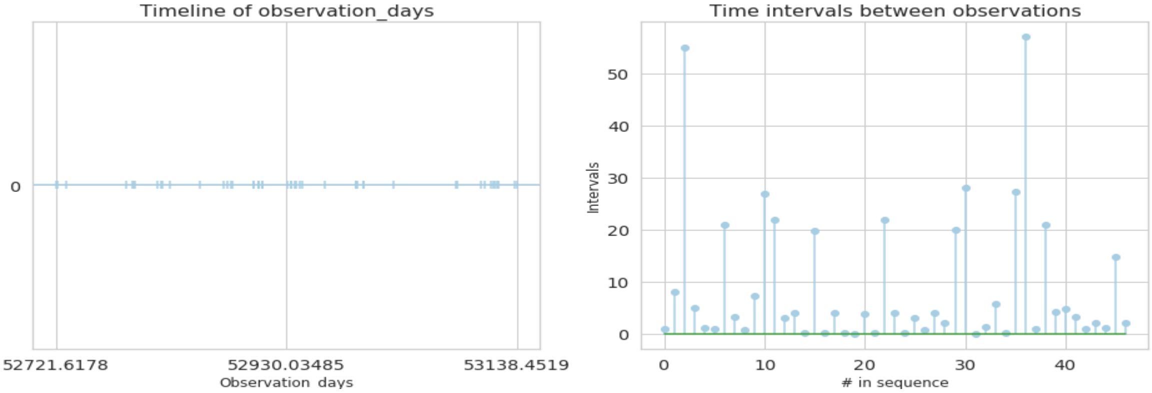

Figure 2 Analysis of observation days

3 Pre-processing of Image Sequences

The goal of pre-processing is to adjust for certain sources of noise that we infer to be present in the data based on our domain knowledge of how the images would be collected with a telescope. The data set that we have is synthetic, but it was created to mimic observation conditions presumed to occur in practice. One of the goals of this project is to analyze how the pre-processing of inputs will affect model training and performance when compared to using sequences of raw images as inputs.

Our pre-processing procedure consisted of two steps. The first step was to remove the ambient light of the sky background that can be seen to vary significantly from night to night. Our method to do this was to subtract the median brightness value, corresponding primarily to the sky background brightness, from each image (adjusting any resultant negative pixel values to 0). This procedure can be easily replicated for pre-processing images collected by a telescope.

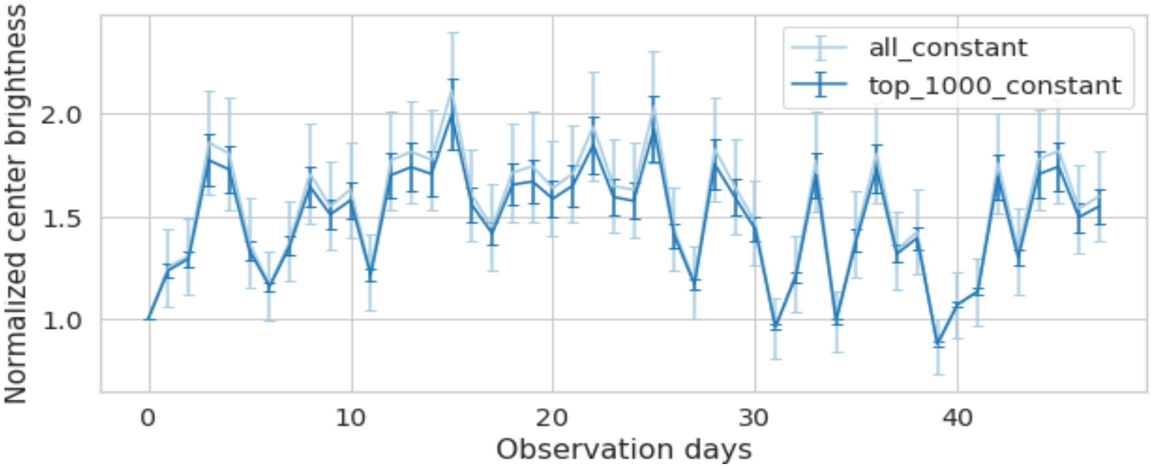

Figure 3

In the second step of pre-processing, we analyzed how the brightness of Constant objects changed within image sequences. Figure 3 shows the brightness of the Constant objects centers (a 5x5 pixel region in our images) for 1000 brightest Constants and also for all Constants in the data set. This figure suggests that the apparent variations in brightness of the Constants are caused by some additional environmental factors with a global component that would affect all objects during a particular night. To adjust for this effect, we found a set of calibration factors that would, on average, even out the brightness of the Constants. These factors were calculated based on the mean central brightness values of the 1000 brightest Constants (which have a better signal-to-noise ratio compared to dimmer Constants) normalized to their brightness during the first night. The factors were then used to adjust the brightness of all image sequences in the data set. For real images from a sky survey conducted on a particular night, this calibration procedure can be performed by using Constants with known locations.

Figure 4 shows how the two pre-processing steps described above greatly reduce the noise for one of the Constant objects in our data set. From the images and the accompanying light curves, we can see that the objects brightness, following pre-processing, appears relatively similar across the whole sequence, as would be expected for a Constant object, such as a non-variable star.

Figures 5, figure 6, figure 7 and figure 8 show the effects of pre-processing on image sequences of six other objects. In all cases, there is a substantial reduction in the noise from the inferred ambient sources, and the signal patterns appear significantly enhanced.

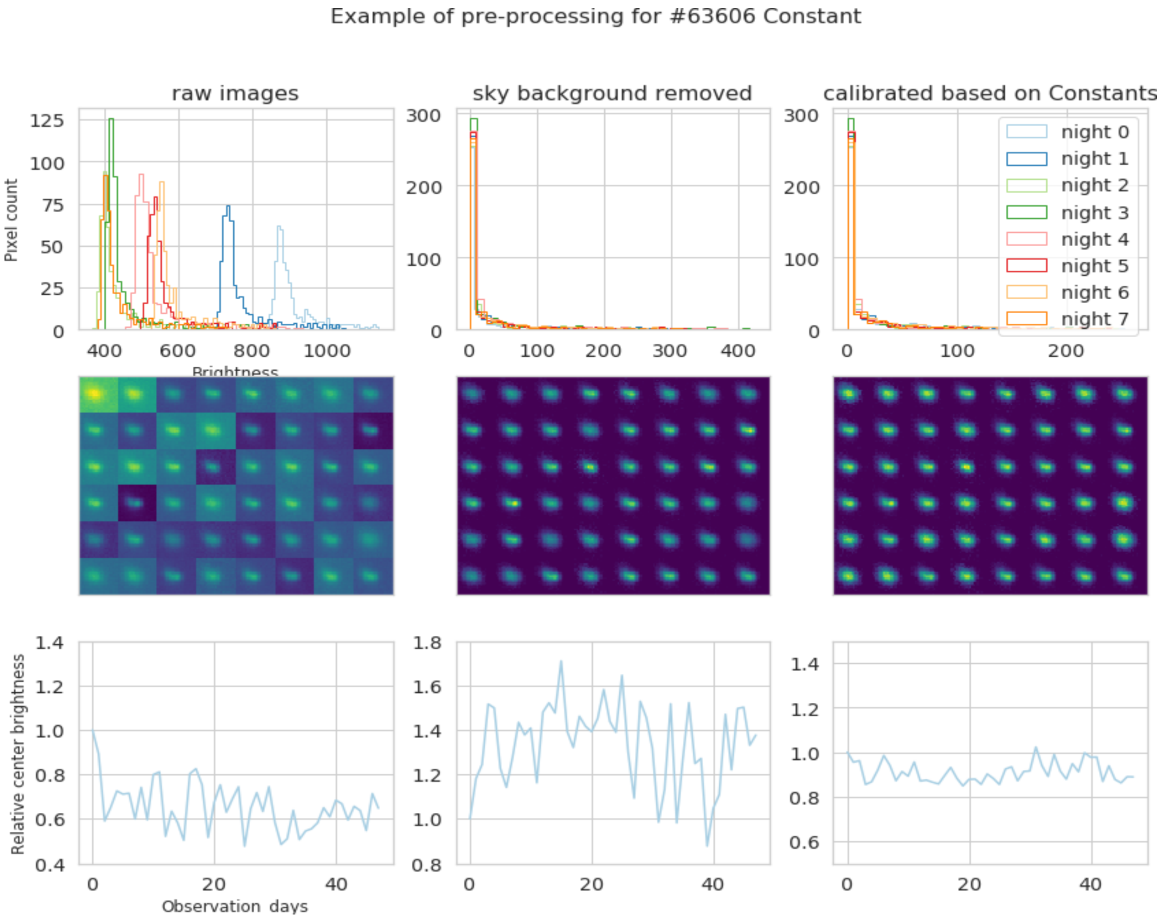
Example of pre-processing for #63606 Constant
raw images
sky background removed
calibrated based on Constants
Pixel count
125
100
75
50
25
0
400
600
800
1000
Brightness
300
200
100
0
0
100
200
300
400
300
200
100
0
0
100
200
night 0
night 1
night 2
night 3
night 4
night 5
night 6
night 7
Relative center brightness
1.4
1.2
1.0
0.8
0.6
0.4
0
20
40
Observation days
1.8
1.6
1.4
1.2
1.0
0.8
0
20
40
1.4
1.2
1.0
0.8
0.6
0
20
40

Figure 4

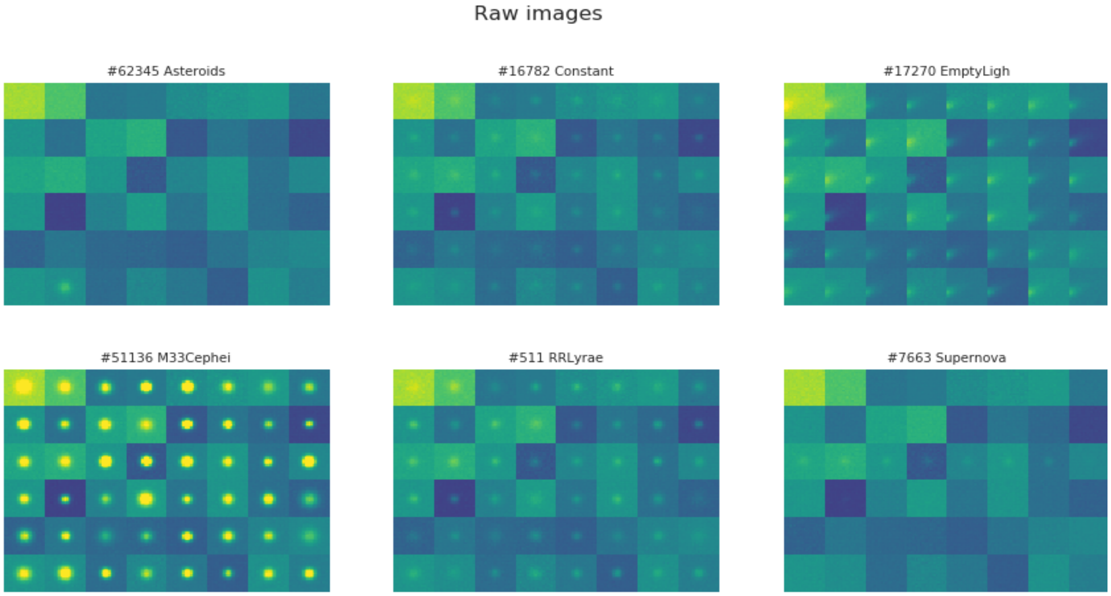

Figure 5

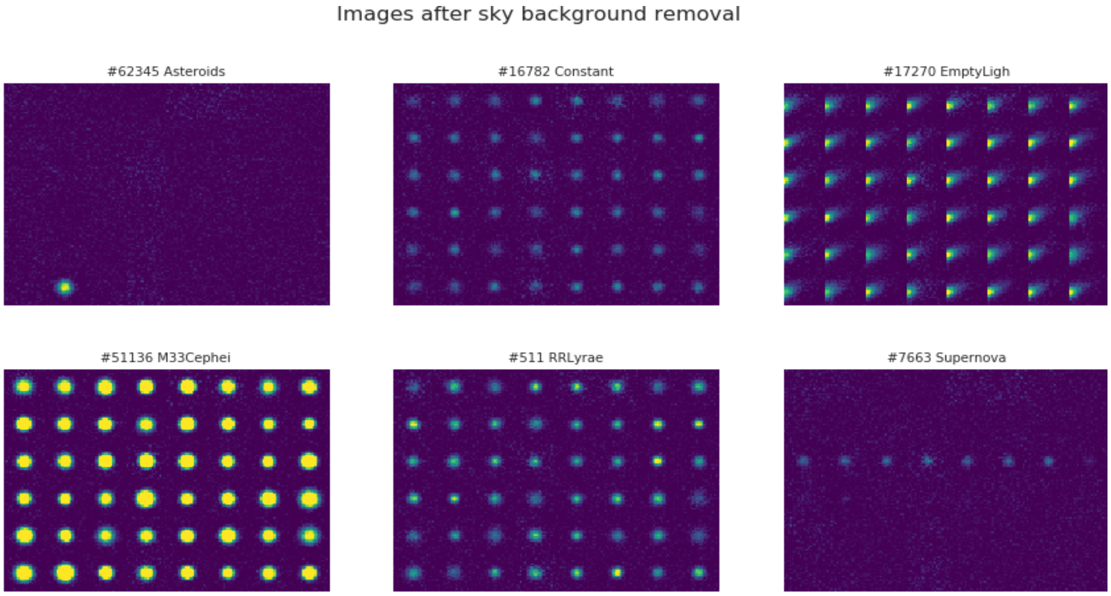

Figure 6

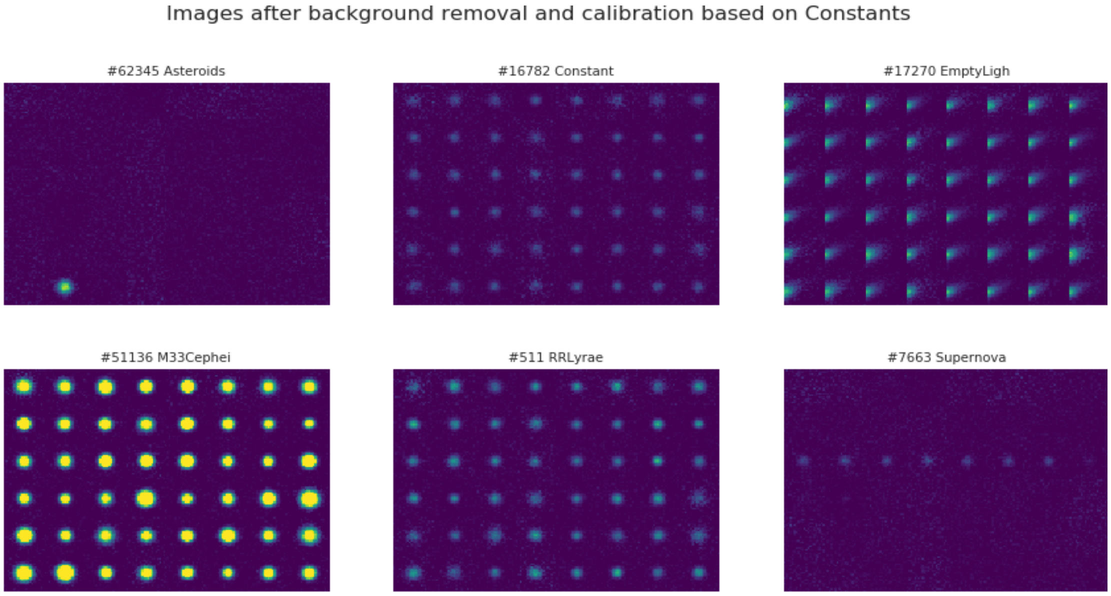

Figure 7

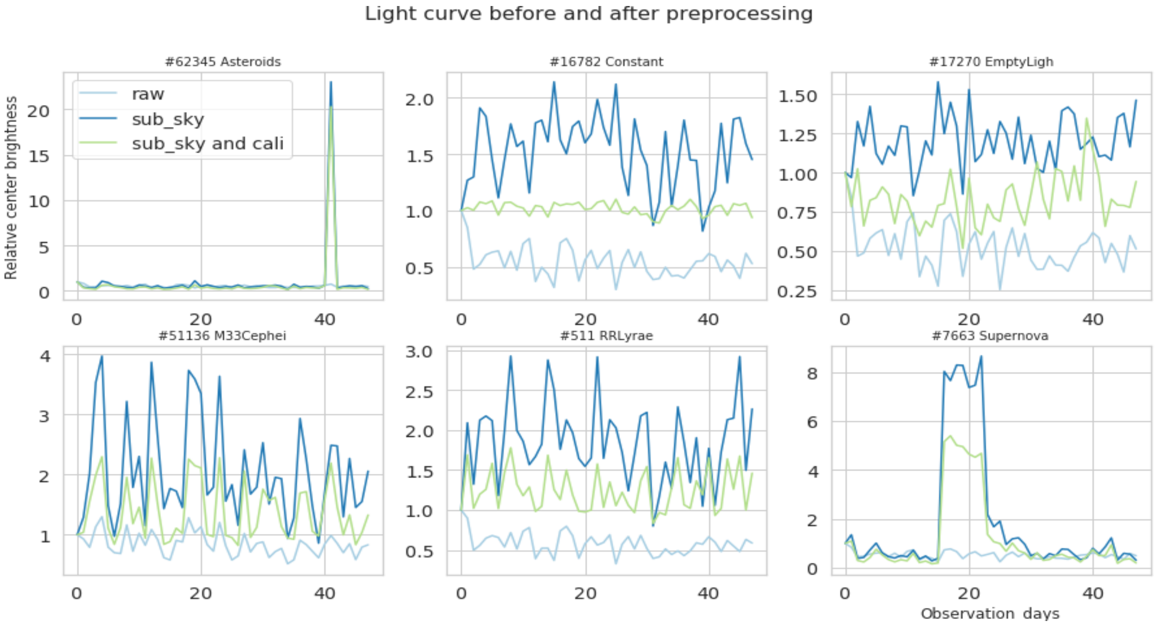

Figure 8

4 Model Architecture

Our data consists of 72,000 sequences of 48 images. In Exploratory Data Analysis we showed that the time intervals between any two images in a sequence vary. However, it is important to note that the set of time intervals $\{t_1 - t_0, t_2 - t_1, \ldots, t_{47} - t_{46}\}$, where t_i is observation day i, is the same for all sequences. In other words, all 72,000 objects are captured at t_0, then at t_1, then at t_2, etc.

Among classes of neural networks, a natural model architecture for encoding image features is a convolutional neural network (CNN), and a natural model architecture for classifying events in a time series is some version of a recurrent neural network (RNN). For this project we combine these two architectures into a RCNN. Conceptually, the CNN portion of the network encodes the images and passes the results sequentially into an RNN that attempts to predict what astronomical event is being represented. As an additional input to the RNN, it is important to include a representation of the time differences between pairs of consecutive images.

Layer	**Layer Prameters**	**Output Dim**
Input Layer	$21 \times 21 \times 1$	$21 \times 21 \times 1$
Conv	$3 \times 3 \times 16$, same	$21 \times 21 \times 16$
Conv	$3 \times 3 \times 16$, same	$21 \times 21 \times 16$
Conv	$3 \times 3 \times 16$, same	$21 \times 21 \times 16$
Max Pooling	2×2, stride 2	$11 \times 11 \times 16$
Conv	$3 \times 3 \times 32$, same	$11 \times 11 \times 32$
Conv	$3 \times 3 \times 32$, same	$11 \times 11 \times 32$
Conv	$3 \times 3 \times 32$, same	$11 \times 11 \times 32$
Max Pooling	2×2, stride 2	$6 \times 6 \times 32$
Fully Connected (with dropout)	1152×512	512
LSTM	$512 + \Delta t$ of samples 512 units	512
Output softmax	512×6	6 (n° classes)

Table 2 RCNN architecture

Astronomical objects in our data set have characteristic light curves that are sampled at different times throughout the image sequences. For an illustrative example of why the time dimension is an important input to the network, consider the fact that the brightness of RR

Lyrae and Cepheid variables varies periodically. These two objects have oscillation frequencies that lie in different characteristic ranges. To correctly interpret the brightness difference in images and to recognize the periodic features specific to RR Lyrae or to Cepheids, it is essential for the model to know how far apart in time the images were taken.

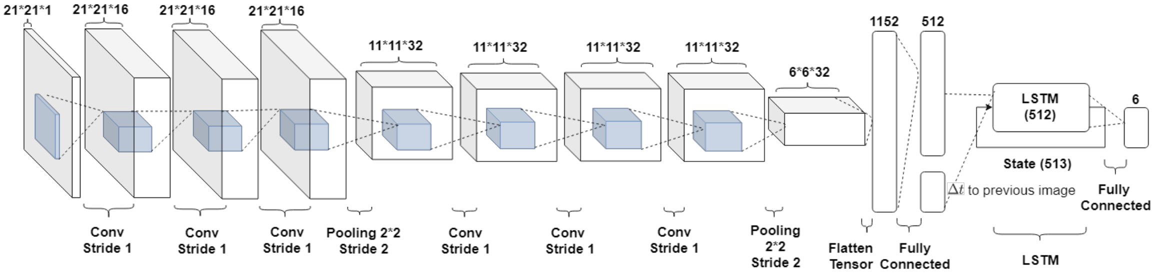

Figure 9 RCNN architecture

Our full model architecture is shown in Table 2, and Figure 9. It was inspired by the design presented in [1], to which we made some modifications. The images in our data set are simple and relatively small. In our preliminary testing we found that a CNN with fewer filters and a smaller Dense layer was able to encode images as well as a larger CNN. Therefore, to speed up training time and to reduce a potential source of overfitting we choose a smaller CNN compared to the one used in [1]. Another difference between the two architectures is that we did not use stacks of images as in [1]. Instead, our methodology focused on passing in the images one-by-one into the LTSM for each sequence.

The model architecture that we used was specified to be able to handle batches with different sizes for the time dimension (# of observations in sequences.) Thus, for instance, it would be possible to train on one batch with sequences that have 48 observations, then train on another batch with sequences that have 30 observations, etc. This feature was important for training one of our model as described in the next section.

5 Training the Models

To investigate the effects of pre-processing and fixed vs varied sequences of observation days as inputs, we trained the following three models:

- **Model A** - RCNN with raw images as inputs and all 48 observation days used during training
- **Model B** - RCNN with pre-processed images as inputs and all 48 observations days used during training
- **Model C** - RCNN with pre-processed images as inputs and randomized sequences of observations days used for each batch during training

For *Model C* we chose the follow procedure to create randomized sequences of observation days for each batch during training:

> **Step 1)** Randomly select the number of observations N that is between between 20 and 47
>
> **Step 2)** Randomly select the starting position, ensuring that there are sufficient observation instances between start and finish for a sequence of N observations
>
> **Step 3)** Randomly sample N observation instances between the start chosen in Step 2 and the last observation day (corresponding to the end of the original sequence). Recalculate the time intervals between observations.

We incorporated Step 2) to specifically randomize the starting location of input sequences for *Model C*, because in our preliminary testing we found that models trained on fixed sequences of observations days focused a lot on the first few starting days and learned how to predict quickly using primarily that information. We hypothesize that the models tended to pay less attention to later days once they got a good a prediction, and, thus, starting from a latter point in the sequence would result in reduced accuracy. A drawback of our methodology for training *Model C* is that it may sometimes not receive some important information about asteroids and supernova. To account for this, we will primarily focus on comparing how well *Model C* classifies non-transient objects compared to *Model A* and *Model B*. For further details, see the the Results and Discussion sections.

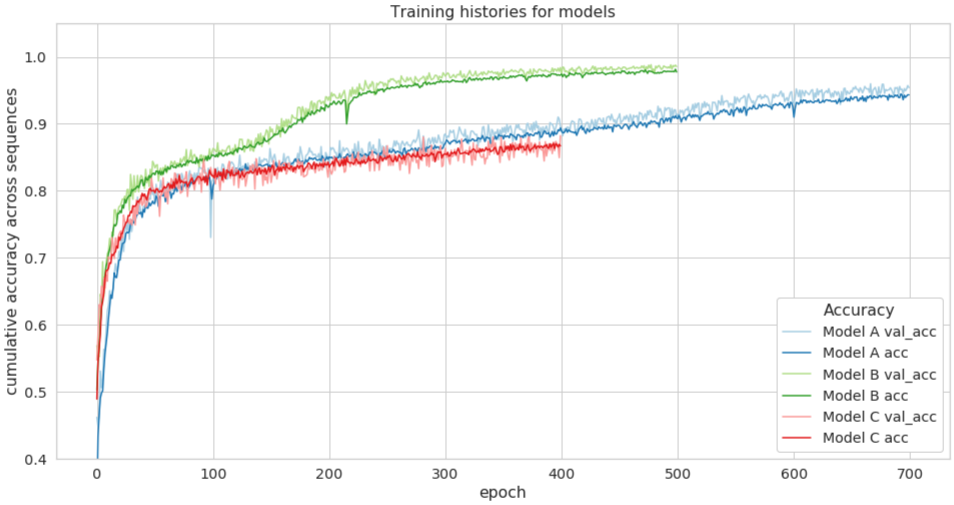

Figure 10

Our models were trained with the AMSGrad variation of the Adam optimizer using batches of size 64. We found 0.0005 to be a good learning rate for our batch size and used these hyperparameters to train all three models. To dynamically generate the batches for *Model C*, we made a generator that performed the random sampling of observation day sequences, as described above. Our RCNN models were many-to-many, producing running predictions at each observation instance in a sequence. These predictions were compared to the actual event labels during a given observation and the loss was calculated using categorical cross-entropy. The total loss function was the sum of the these losses. Thus, the models were rewarded for correctly classifying object classes as early as possible. We trained each model until no improvement in accuracy was apparent within 50 epochs.

Figure 10 shows the training histories of all three models. Pre-processing of images helped *Model B* to reach an overall accuracy (which is a summation of individual observation accuracies) of 0.98 in 500 epochs, while *Model B* reached an overall accuracy of 0.94 in 700 epochs. *Model C* appears to have the lowest overall accuracy. However, *Model C* was trained on sequences with fewer observation days on average, and its overall accuracy shown here is not directly comparable to *Model A* and *Model B*. In the next section, we will show that *Model C* is less overtrained on the same starting location and tends to generalize better.

6 Results and Discussion

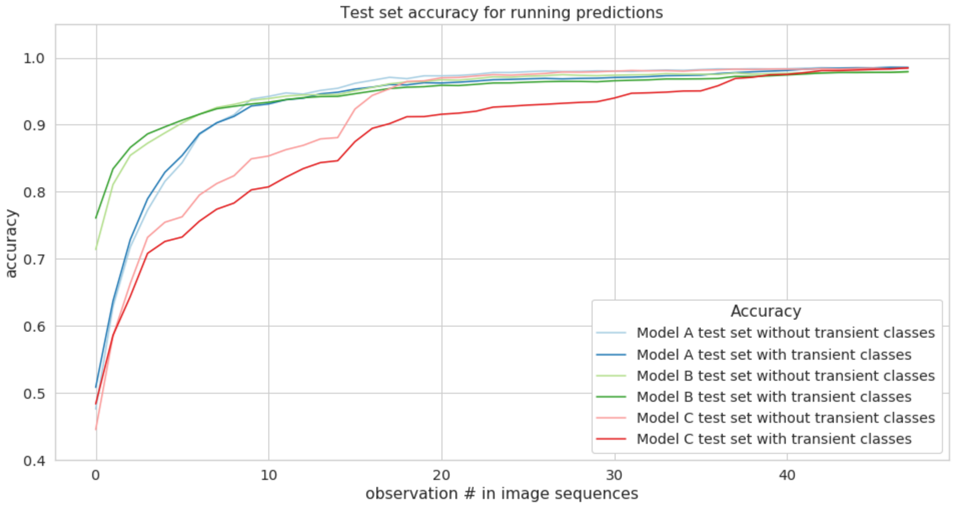

Figure 11

Figure 11 shows how the models performed on a test set of 6000 objects, which were not used during training. These test samples consist of sequences of images corresponding to the full set of 48 observation days. Accuracies reported here indicate how well the models are predicting at each observation day in a sequence. The general trend can be explained by the idea that as the models receive more information they are able to classify better. For this test data, Model A was able to correctly classify objects earlier than the other two models, thus demonstrating the benefit of pre-processing, and of testing the model on the same sequence of observations days as it was trained on.

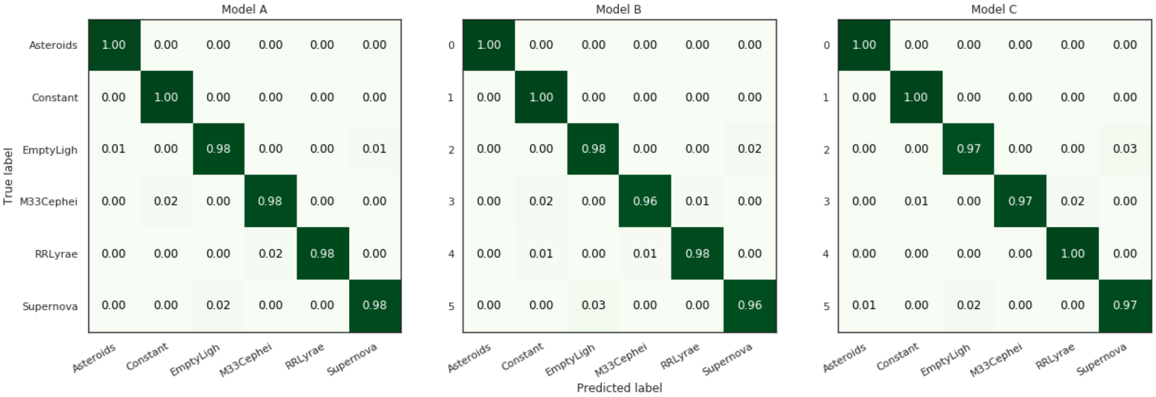

Figure 12 Confusion Matrices

By the final observation day, all three models were able to classify the astronomical objects with very few mistakes as can be seen in Figure 12.

When training a neural network it is important to keep in mind how it will used to make prediction in practice. The data set that we have has a large variety of astronomical objects to be trained on. However, these objects are all imaged at exactly the same dates and times in the 48 observation day sequence. It is natural to suspect that a network trained on the same specific time sequence may end up overfitting to the particular distribution of time intervals that it gets presented over and over again. Such a model may perform poorly in practice when classifying real data that comes with a different sequence of time intervals between observations.

To check whether our *Model A* and *Model B* would generalize to different temporal sequence inputs we conducted two experiments. The first experiment was to look at the performance of our models for 5 rolling windows with a range of 24 consecutive observations as shown in Table 3 Accuracy values were calculated on a test set with the asteroid and supernova events removed. For this experiment, we chose to use only the variable, constant and control objects, because these have a signal signature in every image. Theoretically, for such objects, the amount of information contained in the t_0 to $t_2$3 sequence range would, more or less, be equivalent to the amount of information contained in the $t_2$4 to $t_4$7 sequence range. A more generalized model should be able to perform equally well on both ranges. What we discovered was that the accuracy of *Model A* and *Model B* dropped off substantially as the observation window moved from left to right. We believe that this occurred because *Models*

Observations range	**First three time intervals**	**Model A Accuracy**	**Model B Accuracy**	**Model C Accuracy**
$0 - 23$	$0.97, 7.95, 54.87$	0.98	0.97	0.97
$6 - 29$	$20.93, 3.19, 0.76$	0.85	0.89	0.98
$12 - 35$	$3.02, 4.01, 0.15$	0.55	0.71	0.98
$18 - 41$	$0.19, 0.01, 3.74$	0.55	0.73	0.96
$24 - 47$	$0.15, 3.02, 0.75$	0.51	0.65	0.92

Table 3 *Test set predictions with rolling range windows for observation sequences. Accuracies are computed for the final observation, when the models receive all available information. Asteroids and supernova are excluded from the test set for the accuracy calculations.*

A and *Model B* learned to pay a lot more attention to the early parts of the sequences that they were trained on. As these early parts were sufficient to classify most objects, Models A and B ended up learning not to pay as much attention to the tails of the sequences. It is noteworthy that *Model B* did outperform *Model A*. This may be attributed to the benefit of reducing the noise during pre-processing. We can see that in contrast to *Models A* and *Model B*, *Model C* proved to be robust for all rolling windows achieving very high classification accuracies. This demonstrates the benefit of having more varied data during training.

N_{obs}	**Model A Accuracy**	**Model B Accuracy**	**Model C Accuracy**
40	0.72 ± 0.06	0.96 ± 0.01	0.97 ± 0.01
30	0.64 ± 0.06	0.86 ± 0.07	0.94 ± 0.02
20	0.52 ± 0.03	0.72 ± 0.11	0.88 ± 0.05
10	0.44 ± 0.05	0.64 ± 0.12	0.79 ± 0.05
5	0.42 ± 0.03	0.57 ± 0.17	0.72 ± 0.01

Table 4 *Evaluation of model accuracies on the test set with randomly sampled sequences. Supernova and asteroid objects are excluded from the test set for this analysis. The models make predictions based only on N_{obs} observation occasions that are drawn from the* 48 *available observation days. For each value of N_{obs},* 10 *random drawings are made, and the resulting accuracies are averaged.*

For the second experiment, we looked at model accuracies given sequences of 40, 30, 20, 10 and 5 observations chosen at random. For similar reasons as in the first experiment, only non-transient objects from the test set were used in the analysis. As, can be seen in Table

4 *Model C* once again had the best performance compared to *Model A* and *Model B* for all trials. This serves as further evidence in favor of using more variable time sequences during training. Pre-processing of images appears to be helpful also, given that *Model B* outperformed *Model A* in this experiment.

7 Conclusion

In this project, we analyzed the effects of image pre-processing and found it beneficial to the performance of an RCNN model for classifying astronomical events. We have also shown that, although a neural network can achieve very high accuracy during training, it is important to consider whether there are some aspects of what the network learns that are too specific to the inputs that it sees. In the present work, we showed that models trained on the same temporal sequence of observation days do not generalize well. To mitigate this problem we would propose generating synthetic data with more variation in the time domain, and testing the performance of models on real data, before the models are put into practice. Investigating these strategies would be an interesting extension of the current project.

References

[1] R. Carrasco-Davis, "Deep learning for image sequence classification of astronomical events," *arXiv:1807.03869*, 2018.

[2] N. T., "An optimal extraction algorithm for imaging photometry," *Monthly Notices of the Royal Astronomical Society, 296, 339*, 1998.

[3] M. A. Charnock T., "deep recurrent neural networks for supernovae classification," *The Astrophysical Journal Letters*, 2017.

[4] T. N. Sainath, O. Vinyals, A. Senior, and H. Sak, "Convolutional, long short-term memory, fully connected deep neural networks," *2015 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2015.

[5] v. d. S. P. Zhao R., Ali H., "Two-stream rnn/cnn for action recognition in 3d videos," *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2017.

[6] H. X. Zhao Y., Jin X., "Recurrent convolutional neural network for speech processing," *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2017.