











Image-based Classification of Variable Stars: First Results from Optical Gravitational Lensing Experiment Data

T. Szklenár^{1,2} , A. Bódi^{1,2,3} , D. Tarczay-Nehéz^{1,2} , K. Vida^{1,2,3} , G. Marton^{1,3} , Gy. Mező¹ , A. Forró^{1,2,4} , and R. Szabó^{1,2,3} 

¹ Konkoly Observatory, Research Centre for Astronomy and Earth Sciences, H-1121 Budapest, Konkoly Thege Miklós út 15-17, Hungary; szabo.robort@csfk.mta.hu

² MTA CSFK Lendület Near-Field Cosmology Research Group, Hungary

³ ELTE Eötvös Loránd University, Institute of Physics, Budapest, Hungary

⁴ Eötvös Loránd University, Pázmány Péter sétány 1/A, Budapest, Hungary

Received 2020 April 3; revised 2020 May 29; accepted 2020 June 13; published 2020 July 1

Abstract

Recently, machine learning methods have presented a viable solution for the automated classification of image-based data in various research fields and business applications. Scientists require a fast and reliable solution in order to handle increasingly large amounts of astronomical data. However, so far astronomers have been mainly classifying variable starlight curves based on various pre-computed statistics and light curve parameters. In this work we use an image-based Convolutional Neural Network to classify the different types of variable stars. We use images of phase-folded light curves from the Optical Gravitational Lensing Experiment (OGLE)-III survey for training, validating, and testing, and use OGLE-IV survey as an independent data set for testing. After the training phase, our neural network was able to classify the different types between 80% and 99%, and 77%–98%, accuracy for OGLE-III and OGLE-IV, respectively.

Unified Astronomy Thesaurus concepts: [Astronomy data analysis \(1858\)](#); [Anomalous Cepheid variable stars \(2106\)](#); [Eclipsing binary stars \(444\)](#); [Delta Scuti variable stars \(370\)](#); [RR Lyrae variable stars \(1410\)](#); [Periodic variable stars \(1213\)](#); [Type II Cepheid variable stars \(2124\)](#); [Convolutional neural networks \(1938\)](#); [Sky surveys \(1464\)](#); [Classification \(1907\)](#); [Light curve classification \(1954\)](#)

1. Introduction

Most recent space-borne (e.g., Kepler, see Borucki et al. 2010; Gaia, see Gaia Collaboration et al. 2016; Transiting Exoplanet Survey Satellite (TESS), see Ricker et al. 2014) and ground-based sky surveys (e.g., Sloan Digital Sky Survey (SDSS), see Gunn et al. 2006; and the Vera C. Rubin Observatory (previously referred to as the Large Synoptic Survey Telescope), see Ivezić et al. 2019) generate substantial amounts of data, resulting in new challenges in data processing. This data requires analysis with fast and effective automated computer programming techniques. As a consequence, several machine learning algorithms have become popular in astronomy.

The automatic classification of variable stars using machine learning methods mostly uses photometric data sets where objects are represented by their light curves. The classical approach of variable star classification relies on carefully selected features of the light curves, such as statistical metrics (like mean, standard deviation, kurtosis, skewness; see e.g., Nun et al. 2015), Fourier-decomposition (Kim & Bailer-Jones 2016), or color information (Miller et al. 2015). Classifiers can be trained on manually designed (Pashchenko et al. 2018; Hosenie et al. 2019) or computer-selected features (Becker et al. 2020; Johnston et al. 2020) using known types of variable stars. Another option for classifying light curves utilizes non-labeled data, which is called unsupervised learning. This method clusters similar objects into groups instead of labeling them individually (Mackenzie et al. 2016; Valenzuela & Pichara 2018).

Image-based classification is now a part of our everyday life: we use it in our phones, social network applications, cars, etc. Convolutional neural networks (CNNs; LeCun et al. 1999)—a class of deep neural networks (NNs)—can distinguish between humans, animals, and various objects. If CNNs are well trained,

they can learn very fine features of an image (e.g., face recognition); therefore, this kind of technology is now widely used in many scientific fields such as geology, biology, or even in medicine to recognize tumors and other diseases in the human body (e.g., Alqudah et al. 2020). Recently, CNNs have been successfully applied to astronomical problems as well: real/bogus separation (Gieseke et al. 2017), cold gas study in galaxies (Dawson et al. 2020), supernova classification (Möller & de Boissière 2020), LIGO data classification (George et al. 2018), and exoplanet candidate classification (Osborn et al. 2020). Hon et al. (2018b) trained a convolutional network on 2D images of red giant power spectra to detect solar-like oscillations, and later used the method to classify the evolutionary states of red giants observed by Kepler (Hon et al. 2018a). Carrasco-Davis et al. (2019) designed a recurrent CNN to classify astronomical objects using image sequences; however, their approach does not compute the light curves themselves.

An approach that is similar to ours was used by Mahabal et al. (2017), who transformed the raw light curves into *dmdt* space and mapped the results onto 2D images. These images were then classified using a CNN. Moreover, two other works also took advantage of NNs to classify variables stars. Aguirre et al. (2019) used a recurrent NN that was fed by the light curve measurements one by one as individual points. Aguirre et al. (2019) calculated the difference between consecutive measurement times and magnitude values, and classified the resulted pair of 1D vectors using a CNN. However, the automatic classification of variable stars, which is fully based on the photometric light curves that are represented as images, to our knowledge has not been performed.

In this work, we present the first results of an image-based classification of phase-folded light curves of periodic variable stars with our deep NN architecture trained and validated on the

Optical Gravitational Lensing Experiment (OGLE)-III and tested on OGLE-III and independently on OGLE-IV databases (Udalski et al. 2008, 2015). The goal of our work was to test if we are able to classify the phase-folded light curve images, focusing only on the shape of the light curves and neglecting period information. This idea is very similar to the way human perception works when a traditional astronomer visually evaluates a light curve, i.e., deciding based on distinctive features and patterns. In this study we demonstrate that a deep NN trained with light curve images can effectively be used for classification.

This Letter is structured as follows: in Section 2 we discuss our data selection and handling, in Section 3 we present our NN and data sampling, and in Sections 4 to 6 we show and conclude our results.

2. Data

The aim of our project is to provide an effective and reliable solution for classifying variable stars by means of an image-based classification technique. As a first step, we restrict ourselves to using only periodic variable stars, so that images of phase-folded light curves can be used. Therefore, we need a data set that is classified in a reliable way and contains enough observations to create well-sampled phase-folded light curves.

2.1. Observational Data

Many catalogs of variable stars are available in the literature. Among these, OGLE (Udalski et al. 2015) provides one of the most extensive data sets, as it contains a sufficient number of labels and labeled samples to train and test our NNs. This survey is in its fourth phase, and has been operating since 1992.

OGLE observes the inner Galactic Bulge, the Magellanic Clouds, and the Galactic disk. Observations are obtained in the V - and I -bands; as the latter has about 10 times more data points, we chose to work with the I -band data only.

The obtained light curves mostly have high signal-to-noise ratios and their types are confirmed by experts, which makes the sample very reliable. The OGLE-III catalog lists more than 450,000 variable stars. Along the photometric data, the catalog includes basic parameters of the objects (such as coordinates, periods, mean magnitudes, amplitudes, and parameters of the Fourier light curve decomposition), which can be used to our data preparation process.

The main variable star classes are divided into several subclasses. However, in order to have homogeneous data, we only focused on five different main variable star types observed in the Large Magellanic Cloud (LMC) field during OGLE-III. The chosen types were the following: Anomalous Cepheids (ACep; Soszyński et al. 2008), δ Scuti (DSct; Poleski et al. 2010), eclipsing binaries (ECL; Graczyk et al. 2011), RR Lyrae stars (RRLyr; Soszyński et al. 2009), and Type II Cepheids (T2Cep; Soszyński et al. 2008). The number of objects of each variable type is listed in Table 1.

We converted the measured magnitudes of a given star into flux values with a zero-point of 25, then normalized this data with the maximum brightness. Using the epochs and periods from the OGLE catalog, the light curves have been phase-folded and transformed into 8 bit images with a size of 128×128 pixels, with a black background and white plotted dots (see Figure 1). In case of pulsating variables we used the pulsation periods, while for ECL we used the orbital periods

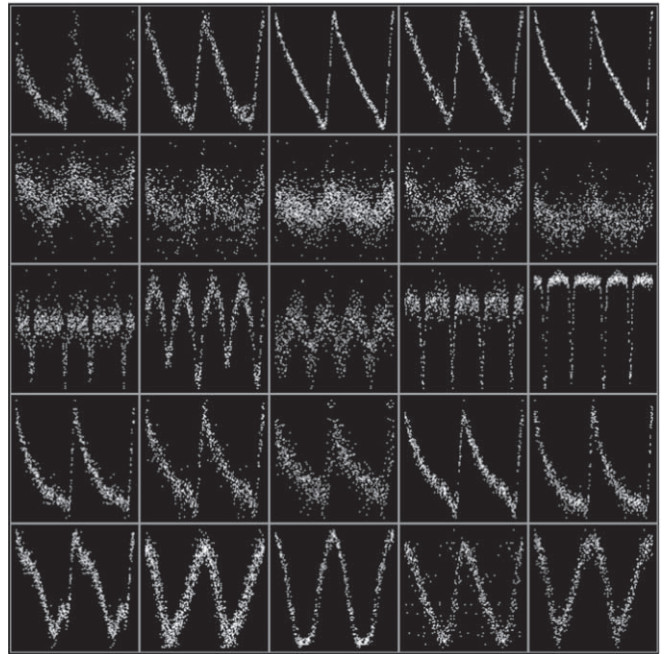


Figure 1. Gallery of phase-folded light curve images of different types of variable stars in OGLE-III. The phases are in the $[0..2]$ interval. From top to bottom: ACep, DSct, ECL, RRLyr, and T2Cep. In case of pulsating variables the light curves are phase-folded by their pulsation periods, while ECL are folded by the orbital periods (i.e., twice the formal periods).

	Non-augmented	Augmented
ACep	83	25,000
DSct	2696	25,000
ECL	26,121	25,000
RRLyr	24,904	25,000
T2Cep	203	25,000
Total	54,007	125,000

(i.e., twice the formal periods) to phase-fold the light curves. Only the raw data were used, without sigma clipping and measurement error handling. In order to ensure that all of the representative light curve shapes were covered, the phased light curves are plotted in the $[0..2]$ phase interval. These images served as the basis of our training sample.

One other purpose of our research was to know how well our trained model works with other observational data. This is why we generated light curves from the OGLE-IV database. Unfortunately, δ Scuti stars have not been published yet, so we only used ACep (Soszyński et al. 2015), ECL (Pawlak et al. 2016), RRLyr (Soszyński et al. 2016), and T2Cep (Soszyński et al. 2018). The subtypes were not separated, and we used the same method for image generation.

2.2. Data Augmentation

A highly unbalanced number of representatives in different classes, which is the case here (as can be seen in Table 1), may cause false machine learning output; therefore, data augmentation was crucial in the pre-processing phase. The basic data augmentation methods usually use, e.g., Gaussian noise, elastic transform, random brightness, or contrast changes (see, e.g.,

Shorten & Khoshgoftaar 2019); the images can also be mirrored or rotated. These methods allow us to create more duplicates, and work well when classifying everyday objects: the reflection of a cat in a mirror is still a cat—however, this is not true for light curves.

To increase the sample of under-represented classes, randomly generated noise was sampled from a Gaussian distribution with zero mean and standard deviation equal to the given measured error then added to the original light curves. The augmented training data contained 125,000 images, 25,000 from each variable star type (the number of ECL was reduced). We took precautions to ensure that the training, testing, and validating sets contained non-overlapping samples of the generated dummy light curves, i.e., for one given star the original light curve and its generated dummy multiples were used only in one of the aforementioned steps.

3. Methods and Models

3.1. DarkNet

In order to investigate the effectiveness of a CNN on classifying the folded light curves, first we tested DarkNet (Redmon 2013–2016), a GPU-supported open-source software. We used a built-in, very simple convolutional NN for the first training of our data. This was originally created for the CIFAR-10 data set,⁵ which is a test to classify images from 10 different classes of freely downloadable 28×28 pixel images of cars, dogs, cats, ships, etc.

Our first training package was not augmented, so the classes had varying amounts of data. This set contained 54,007 images; see Table 1.

Training a deep NN requires a vast amount of computational time and capacity. To be able to test our first deep NN on simple desktop computers, we created a much smaller image package. The first training was conducted with fewer than 500 images, but it took 1.5 hr to complete it. We used a high-performance computer, containing four Tesla V100 GPUs, for this task. Although the first results using the DarkNet framework were promising, due to the poor documentation the complexity of architecting a network and the required time to preprocess data to match the format with the DarkNet requirements, we decided to move to a more user-friendly framework.

3.2. TensorFlow/Keras

As the DarkNet package is poorly documented and not being maintained, we compiled a new NN based on the TensorFlow/Keras framework. TensorFlow (Abadi et al. 2015) is a free and open-source framework that is widely used in different machine learning applications. Keras (Chollet et al. 2018) is an open-source, high-level language and NN library for creating deep NNs with ease and has been officially supported in the TensorFlow core library since 2017. As a first step we recreated the previous CNN, now in Keras, using Python programming language. During the testing phase we used TensorBoard (Abadi et al. 2015) to visualize the differences and track the changes in training loss and accuracy.

3.3. Our CNN

Convolutional networks use convolution instead of general matrix multiplication in their layers. A typical network architecture uses a mixture of convolutional, pooling, and fully connected layers. Additionally, dropout layers can be added for regularization purposes. CNNs set the weights for the filter kernels during the learning process instead of using pre-set kernels as in, e.g., early optical character recognition solutions: this independence from prior knowledge gives them great flexibility and the ability to recognize features on different spatial scales in their consecutive layers.

Figure 2 shows a schematic view of our CNN. Our model has a conventional structure: it consists of two convolutional, one dropout, and one pooling layer, in all four blocks. The resolution of input images is 128×128 pixels; the first two convolutional layers use a 16×16 pixels width convolutional window (known as “kernel/filtersize”), with 1 pixel stride to run through the images. After this step, the second, third, and fourth pair of convolutional layers use 8×8 , 4×4 , and 2×2 pixel wide windows, respectively. This way, our model can learn the low-level features in the beginning of the training process and the high-level features during the last convolutional layers as well. All convolutional layers use Rectified Linear Unit (ReLU) activation.⁶ The output of the last convolution block is flattened and sent to a network of fully connected layers (dense layers). The last one is a softmax layer, which is used to normalize the output and hence yields predictions (numbers between 0 and 1) for all the five possible output labels. All together, the total number of trainable parameters is 1,615,685. The tested hyperparameters and the final chosen ones are listed in Table 2.

3.3.1. Convolutional Layers

The input for the convolutional layer is a tensor with the shape of the image height, width, and depth. When data is passing through this layer it becomes abstracted by a feature map. During this step a filter matrix—or kernel—of a given size is convolved with parts of the image by moving it with a given stride until the whole image is traversed. These layers can detect low-level features in the first steps, but can extract high-level features in later stages.

3.3.2. Pooling Layers

The pooling layer is responsible for reducing the spatial size of the convolved image. It reduces the required computational power and is also important for the extraction of dominant features of the image. We used maximum pooling in our model, which returns the maximum value from each portion of an image: it selects important features as well as reduces noise.

3.3.3. Fully Connected Layers

Fully connected layers (also known as dense layers) are responsible for the classification process as they can learn the nonlinear combinations of the high-level features represented by the convolutional layers. As a final step, we use a softmax classification (basically a generalized version of a sigmoid function for multiple outputs), which classifies our images into separate classes of variable stars.

⁵ <https://www.cs.toronto.edu/~kriz/cifar.html>

⁶ $f(x) = \max(0, x)$.

Table 3
Number of Images Used in the Various Steps

Survey	Training	Validation	Testing
OGLE-III	87,500	18,750	18,750
OGLE-IV	10,000

3.7. Evaluation Metrics

The performance of a trained machine learning algorithm can be quantitatively characterized through several evaluation metrics. The metric where the input and predicted class labels are plotted against each other is called a confusion matrix, where the predicted class is indicated in each column and the actual class in each row. This method allows us to visualize the number of true/false positives and negatives. In the best-case scenario, if the matrix is normalized to unity, we expect the confusion matrix to be purely diagonal, with non-zero elements on the diagonal, and zero elements otherwise.

Precision is defined as

$$\text{Precision} = \frac{\text{TP}}{\text{TP} + \text{FP}}, \quad (1)$$

where TP is the number of true positives and FP is the number of false positives. Precision shows that how precise the final model is out of those predicted positive, i.e., how many of predicted positives are actual positive.

Recall is defined as

$$\text{Recall} = \frac{\text{TP}}{\text{TP} + \text{FN}}, \quad (2)$$

where TP is the number of true positives and FN is the number of false negatives. Recall shows how many of the actual positives are labeled by the model as true positives.

From the last two metrics the F1 score can be calculated, which is the harmonic average of the precision and recall:

$$\text{F1} = 2 \cdot \frac{\text{Precision} \cdot \text{Recall}}{\text{Precision} + \text{Recall}}. \quad (3)$$

The F1 score can measure the accuracy of the model, which returns a value between 0 and 1, where the latter corresponds to a better model.

4. Results

4.1. Training and Validation

Our final data set contained 125,000 images, 25,000 from each type. This data set was subdivided into three different parts (70%–15%–15%), choosing images without any overlap for training, validation, and testing purposes, respectively. The number of images used for training was 87,500, and 18,750 were used for validation (see Table 3). The process that goes through these two phases (training and validation) is called an epoch. We used GPU-accelerated computers provided by the MTA Cloud⁷ and the Konkoly Observatory for this research. Each training and validation epoch took about 290 s on a NVidia Tesla K80 GPU-supported computer and 62 s with a NVidia GeForce RTX 2080 Ti GPU card. Accuracy and loss values were constantly monitored. An EarlyStopping callback

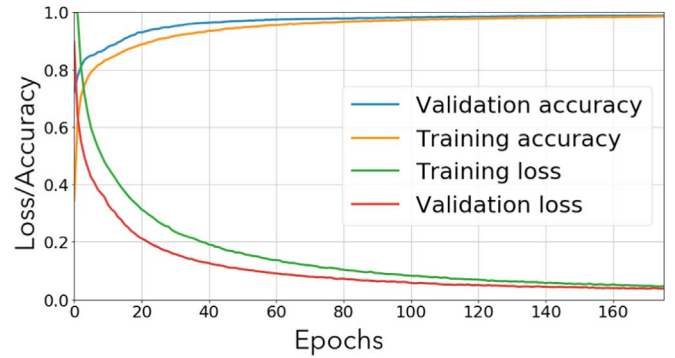


Figure 3. Accuracy and loss of the training and validation process. Orange curve: training accuracy, red curve: validation accuracy. Blue curve: training loss, green curve: validation loss.

	ACEP	DSCT	ECL	RRLYR	T2CEP
ACEP	80.2	0.6	0.0	17.3	1.8
DSCT	0.2	94.0	0.8	4.7	0.4
ECL	0.0	1.1	98.8	0.1	0.1
RRLYR	0.9	2.7	0.0	95.8	0.5
T2CEP	1.2	0.7	0.0	18.6	79.5

Figure 4. Test results from the OGLE-III data.

stopped the training process after 173 full epochs. Inspecting the log files in TensorBoard showed no overfitting, after epoch 173 we reached 98.5% training accuracy for the complete model (see Figure 3).

4.2. Testing the Model

We made two separate prediction tests on our model. The first one ran on the previously mentioned OGLE-III data.

Our original data set was divided randomly into three different parts: 87,500 images were used for training, 18,750 for validation, and the remaining 18,750 light curve images were for testing purposes. This test data set contained 3750 images from each variable star type and the test method ran through all light curves, using the weights from our trained model. We received a predicted label for each image, and at the end of the test we could see how well our model was working with the OGLE-III LMC data (see Figure 4).

For our second test we generated 10,000 augmented samples (2500 from each type) from the OGLE-IV database (see Table 3). The method was the same as before: we made predictions on each image, using the weights from the trained

⁷ <https://cloud.mta.hu>

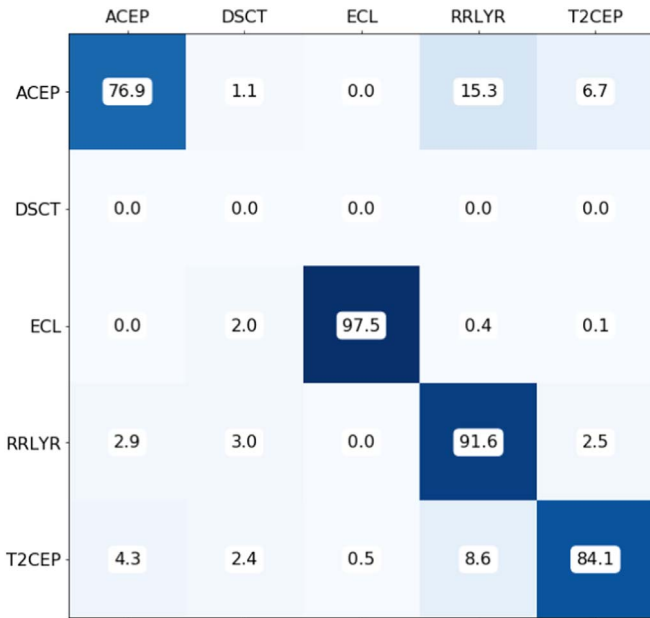


Figure 5. Test results from the OGLE-IV data.

network (see Figure 5 and Table 5). Comparing the two confusion matrices, it is clearly visible that our trained model works well and can classify variable stars from a different database.

Tables 4 and 5 show that a CNN trained on phase-folded light curves can classify variable stars with very high accuracy. Based on our results, we conclude that our model can efficiently distinguish between the ECL and periodic variables like RR Lyrae stars. However, we note that due to similar light curve shapes and noise features, and due to the fact that we refrained from using the period as an input parameter, we received false predictions for the pulsating stars in our second test. ACep, RRLyr, and T2Cep stars were especially vulnerable to false prediction, while DSct stars were not available for OGLE-IV, as previously mentioned.

In this research we focused only on the light curve shapes, but it will be possible in the future to insert other data (most importantly, the period) into a more complex multi-channel network that could handle more inputs, image, and numerical data as well.

5. Discussion

Our method uses a relatively new approach to classify the light curves of periodic variable stars; unlike similar NNs, we neither use the time stamps of the measurements directly nor do we transform the light curves into another space. Instead, we look only at the light curve shape characteristics, which is achieved by phase-folding to increase the sampling within a cycle to be able to describe the shape more precisely. To compare our results with a more traditional method, we trained an RF algorithm using amplitudes and R_{21} , ϕ_{21} Fourier-parameters that best characterize the light curve shapes. Comparing Figure 4 to Table 4, and Figure 5 to Table 5, i.e., OGLE-III and OGLE-IV results, respectively, we can see that overall our CNN algorithm predicts better. Two cases where the CNN performs worse are the OGLE-III ACep and T2Cep classes; however, as we conduct a transfer learning, i.e., test these methods on the independent OGLE-IV data set, we find

Table 4
Classification Report for the Five Classes in the OGLE-III Data Set

	Precision	Recall	F1 Score
ACep	0.803/0.93	0.972/0.95	0.879/0.94
DSct	0.939/0.87	0.949/0.89	0.944/0.88
ECL	0.987/0.98	0.992/0.95	0.989/0.96
RRLyr	0.959/0.88	0.702/0.86	0.810/0.87
T2Cep	0.795/0.96	0.966/0.96	0.872/0.96
Average	0.897/0.92	0.916/0.92	0.899/0.92

Note. Numbers correspond to the CNN and RF trainings, respectively. The confusion matrix for this report is shown in Figure 4.

Table 5
Classification Report for the Five Classes in the OGLE-IV Data Set

	Precision	Recall	F1 Score
ACep	0.769/0.89	0.914/0.62	0.835/0.73
DSct
ECL	0.975/0.96	0.994/0.94	0.984/0.95
RRLyr	0.916/0.72	0.790/0.90	0.849/0.80
T2Cep	0.841/0.99	0.900/0.75	0.870/0.85
Average	0.875/0.89	0.900/0.80	0.885/0.83

Note. Numbers correspond to the CNN and RF trainings, respectively. The confusion matrix for this report is shown in Figure 5.

that our CNN gives similar results as before, while RF gives worse results by 15%–16% for the mentioned classes.

The quality of training sets is an important aspect to note. As described in Section 2, we did not clear our sample, i.e., we included outliers and low-quality data, which makes the training more realistic and a harder task for the CNN to learn the weights. However, these bad values have a subtle impact on the calculation of Fourier-parameters, making the RF result more boosted.

Anomalous Cepheids are relatively larger mass ($1\text{--}2 M_{\odot}$) variable stars that lie in the classical instability strip. They follow a period–luminosity relation, and their luminosity is between that of classical and Type II Cepheids. They are pulsating in the fundamental mode or the first overtone with a period shorter than 2 days. Their light curve is characterized by a steeper ascending branch that is followed by a shallower descending branch. Usually a bump is present at the bottom of the ascending branch. These features make it very hard or nearly impossible to distinguish them from RR Lyrae stars without known distances, i.e., their absolute brightness.

One of the main goals of our work is to see whether a CNN can distinguish Anomalous Cepheids from other variable stars based only on light curve characteristics. From Figure 4 and Table 6 we can see that our CNN was able to well classify the 80.2%, while the RF, which is based on pre-computed features, well classified the 95.46% of ACeps in the OGLE-III sample. As expected, the majority of the misclassifications are labeled as RR Lyrae stars (17.3% and 4.5%). These results show that there are hints of differences that make it possible to separate ACeps without known distances. Regarding our work, it is interesting that the CNN classifies ACeps about 15% worse than the RF. However, if we test these methods using the independent OGLE-IV database (see Figure 5 and Table 5), our

Table 6

Classification Report for the Five Classes in the OGLE-III Data Set of the RF Method

		Predicted Class				
		ACep	DSct	ECL	RRLyr	T2Cep
True class	ACep	95.46	0.08	0.00	4.46	0.00
	DSct	0.86	89.26	1.85	5.46	2.57
	ECL	0.00	3.91	95.11	0.14	0.85
	RRLyr	4.50	7.85	0.42	86.45	0.77
	T2Cep	2.12	0.89	0.00	1.46	95.53

CNN still performs near 80% (76.9%), while the performance of RF drops down to 62% (see Table 7). However, this decrease is not entirely surprising, as RFs are restricted to predict within the range of input parameters, i.e., they are not useful for transfer learning.

These results mean that image-based CNN classification may take place in applications where the training set slightly differs from the data set on which the prediction will be made.

6. Conclusions and Future Prospects

In this work we trained a deep NN to distinguish different types of variable stars based on light curve images generated from the OGLE-III database. To be able to do this, we generated a data-augmented image data set, containing equal amounts of images from the chosen five types of variable stars. After thorough testing, a CNN was created that learned the different light curve features with high level of accuracy.

We demonstrated that image-based variable star classification is a viable option using a CNN. This type of machine learning method can learn both the high- and low-level features of a folded variable starlight curve (explanation: the variable star, i.e., a special class of stars, has a light curve, i.e., the change in brightness as a function of time) with high level of accuracy, in our case between 80% and 99%, based on OGLE-III data. It is clearly visible from our results that additional data (e.g., period) could increase the classification accuracy. We are working on a multi-channel network where additional important parameters can be added as input; this way, we expect that the classification accuracy of different variable star types will continue to increase. Our future plans also include generating light curve images for all variable stars in the OGLE-III LMC and Small Magellanic Cloud fields using their subtypes available (e.g., RRab/RRc/RRd instead of RRLyr) as well. This way we would have a vast amount of training data, and our model could be more specific and reliable.

Training and testing a CNN requires vast amount of computational time and capacity. We used GPU-accelerated computers in this research. However, making predictions (i.e., classification itself) is possible with the saved weight file on any commercial computers. Predicting a label for one image takes just a fraction of a second (e.g., 0.13 s on a simple laptop), meaning that predictions even on large amount of light curve images can be made in a very short time (see Table 8).

A novel way of variable star classification would make a difference in the interpretation of the billions of light curves available today (and more to come). The Zwicky Transient Facility (ZTF; Masci et al. 2018) produced ~ 1 billion light curves with more than 20 data points at different epochs, and this number is growing continually. The All-Sky Automated Survey for Supernovae (ASAS-SN; Shappee et al. 2014;

Table 7

Classification Report for the 5 Classes in the OGLE-IV Data Set of the RF Method

		Predicted Class				
		ACep	DSct	ECL	RRLyr	T2Cep
True class	ACep	62.02	9.41	0.00	28.57	0.00
	DSct
	ECL	0.00	4.92	94.06	0.14	0.89
	RRLyr	4.62	5.16	0.07	89.92	0.24
	T2Cep	3.03	11.67	4.20	6.52	74.57

Table 8

Approximate Computational Runtimes in Minutes

Survey	Preprocess	Training	Testing
OGLE-III 125,000 images	265	167/61	28/12
OGLE-IV 10,000 images	30	...	23/5

Note. Numbers correspond to the NVidia Tesla K80, and NVIDIA RTX 2080 Ti GPU cards, respectively.


Kochanek et al. 2017) database currently contains 61.5 million light curves, out of which 666,500 objects were found to be variables. At least 62,500 of them have unreliable classifications. The First Catalog of Variable Stars Measured by ATLAS (ATLAS-VAR; Heinze et al. 2018) has already detected 4.7 million variable objects, but only 214,000 of them received specific classifications. TESS (Ricker et al. 2015) is in its second year of operations and keeps collecting excellent quality data from space with high cadence, like Gaia (Gaia Collaboration et al. 2016) does for billions of sources on the entire sky with lower cadence, and only a small fraction of them is classified accurately (below 1%, see e.g., Molnár et al. 2018; Gaia Collaboration et al. 2019; Marton et al. 2019). Future surveys, like the Rubin Observatory Legacy Survey of Space and Time (LSST; Ivezić et al. 2019) will further increase the number of objects with available time series data. One can see that astronomy needs accurate and efficient methods to rapidly analyze and classify variable objects in the sky. In the upcoming papers of this series we will explore further light curve data with the ultimate goal of providing such methods using image-based classification.

This project has been supported by the Lendület Program of the Hungarian Academy of Sciences, project No. LP2018-7/2019, the NKFI KH-130526 and NKFI K-131508 grants, the Hungarian OTKA grant No. 119993, and the MW-Gaia COST Action (CA 18104). G.M. was supported by the Hungarian National Research, Development and Innovation Office (NKFIH) grant PD-128360. G.M. acknowledges partial support from the EC Horizon 2020 project OPTICON (730890) and the ESA PRODEX contract No. 4000129910. This project has received funding from the European Research Council (ERC) under the European Union’s Horizon 2020 research and innovation program under grant agreement No. 716155 (SACCRED). On behalf of “*Analysis of space-borne photometric data*” project we thank for the usage of MTA Cloud (<https://cloud.mta.hu>) that significantly helped us achieving the results published in this Letter. The authors thank the

referee for helpful comments that significantly improved the manuscript.

Software: Python (van der Walt et al. 2011), Numpy (van der Walt et al. 2011), Pandas (McKinney 2010), Scikit-learn (Pedregosa et al. 2011), Tensorflow (Abadi et al. 2015), Keras (Chollet et al. 2018).

ORCID iDs

T. Szklenár  <https://orcid.org/0000-0002-5610-7697>
 A. Bódi  <https://orcid.org/0000-0002-8585-4544>
 D. Tarczay-Nehéz  <https://orcid.org/0000-0003-3759-7616>
 K. Vida  <https://orcid.org/0000-0002-6471-8607>
 G. Marton  <https://orcid.org/0000-0002-1326-1686>
 Gy. Mező  <https://orcid.org/0000-0002-0686-7479>
 A. Forró  <https://orcid.org/0000-0001-9394-3531>
 R. Szabó  <https://orcid.org/0000-0002-3258-1909>

References

- Abadi, M., Agarwal, A., Barham, P., et al. 2015, TensorFlow: Large-Scale Machine Learning on Heterogeneous Systems, <https://www.tensorflow.org/>
- Aguirre, C., Pichara, K., & Becker, I. 2019, *MNRAS*, **482**, 5078
- Alqudah, A. M., Alquraan, H., Abu Qasmieh, I., Alqudah, A., & Al-Sharu, W. 2020, arXiv:2001.08844
- Becker, I., Pichara, K., Catelan, M., et al. 2020, *MNRAS*, **493**, 2981
- Borucki, W. J., Koch, D., Basri, G., et al. 2010, *Sci*, **327**, 977
- Breiman, L. 2001, *Machine Learning*, **45**, 5
- Carrasco-Davis, R., Cabrera-Vives, G., Förster, F., et al. 2019, *PASP*, **131**, 108006
- Chollet, F. 2018, Keras: The Python Deep Learning Library, Astrophysics Source Code Library, ascl:1806.022
- Dawson, J. M., Davis, T. A., Gomez, E. L., et al. 2020, *MNRAS*, **491**, 2506
- Gaia Collaboration, Eyer, L., Rimoldini, L., et al. 2019, *A&A*, **623**, A110
- Gaia Collaboration, Prusti, T., de Bruijne, J. H. J., et al. 2016, *A&A*, **595**, A1
- George, D., Shen, H., & Huerta, E. A. 2018, *PhRvD*, **97**, 101501
- Gieseke, F., Bloemen, S., van den Bogaard, C., et al. 2017, *MNRAS*, **472**, 3101
- Graczyk, D., Soszyński, I., Poleski, R., et al. 2011, *AcA*, **61**, 103
- Gunn, J. E., Siegmund, W. A., Mannery, E. J., et al. 2006, *AJ*, **131**, 2332
- Heinze, A. N., Tonry, J. L., Denneau, L., et al. 2018, *AJ*, **156**, 241
- Hon, M., Stello, D., & Yu, J. 2018a, *MNRAS*, **476**, 3233
- Hon, M., Stello, D., & Zinn, J. C. 2018b, *ApJ*, **859**, 64
- Hosenie, Z., Lyon, R. J., Stappers, B. W., & Mootooyaloo, A. 2019, *MNRAS*, **488**, 4858
- Johnston, K. B., Caballero-Nieves, S. M., Petit, V., Peter, A. M., & Haber, R. 2020, *MNRAS*, **491**, 3805
- Ivezić, Ž., Kahn, S. M., Tyson, J. A., et al. 2019, *ApJ*, **873**, 111
- Kim, D.-W., & Bailer-Jones, C. A. L. 2016, *A&A*, **587**, A18
- Kochanek, C. S., Shappee, B. J., Stanek, K. Z., et al. 2017, *PASP*, **129**, 104502
- LeCun, Y., Haffner, P., Bottou, L., & Bengio, Y. 1999, in Feature Grouping, ed. D. Forsyth (Berlin: Springer)
- Mackenzie, C., Pichara, K., & Protopapas, P. 2016, *ApJ*, **820**, 138
- Mahabal, A., Sheth, K., Gieseke, F., et al. 2017, arXiv:1709.06257
- Marton, G., Ábrahám, P., Szegedi-Elek, E., et al. 2019, *MNRAS*, **487**, 2522
- Masci, F. J., Laher, R. R., Rusholme, B., et al. 2018, *PASP*, **131**, 018003
- McKinney, W. 2010, in Proc. 9th Python in Science Conf., SciPy 2010, ed. S. van der Walt & J. Millman, 51
- Miller, A. A., Bloom, J. S., Richards, J. W., et al. 2015, *ApJ*, **798**, 122
- Möller, A., & de Boissière, T. 2020, *MNRAS*, **491**, 4277
- Molnár, L., Plachy, E., Juhász, Á. L., & Rimoldini, L. 2018, *A&A*, **620**, A127
- Nun, I., Protopapas, P., Sim, B., et al. 2015, arXiv:1506.00010
- Osborn, H. P., Ansdell, M., Ioannou, Y., et al. 2020, *A&A*, **633**, A53
- Pashchenko, I. N., Sokolovsky, K. V., & Gavras, P. 2018, *MNRAS*, **475**, 2326
- Pawlak, M., Soszyński, I., Udalski, A., et al. 2016, *AcA*, **66**, 421
- Pedregosa, F., Varoquaux, G., Gramfort, A., et al. 2011, Journal of Machine Learning Research, 2011, hal-00650905, <http://hal.inria.fr/hal-00650905>
- Poleski, R., Soszyński, I., Udalski, A., et al. 2010, *AcA*, **60**, 1
- Redmon, J. 2013–2016, Darknet: Open Source Neural Networks in C, <http://pjreddie.com/darknet/>
- Ricker, G. R., Winn, J. N., Vanderspek, R., et al. 2014, *Proc. SPIE*, **9143**, 914320
- Ricker, G. R., Winn, J. N., Vanderspek, R., et al. 2015, *JATIS*, **1**, 014003
- Shappee, B. J., Prieto, J. L., Grupe, D., et al. 2014, *ApJ*, **788**, 48
- Shorten, C., & Khoshgoftaar, T. M. 2019, *Journal of Big Data*, **6**, 60
- Soszyński, I., Udalski, A., Szymański, M. K., et al. 2008, *AcA*, **58**, 293
- Soszyński, I., Udalski, A., Szymański, M. K., et al. 2009, *AcA*, **59**, 1
- Soszyński, I., Udalski, A., Szymański, M. K., et al. 2015, *AcA*, **65**, 233
- Soszyński, I., Udalski, A., Szymański, M. K., et al. 2016, *AcA*, **66**, 131
- Soszyński, I., Udalski, A., Szymański, M. K., et al. 2018, *AcA*, **68**, 89
- Udalski, A., Szymanski, M. K., Soszynski, I., & Poleski, R. 2008, *AcA*, **58**, 69
- Udalski, A., Szymański, M. K., & Szymański, G. 2015, *AcA*, **65**, 1
- Valenzuela, L., & Pichara, K. 2018, *MNRAS*, **474**, 3259
- van der Walt, S., Colbert, S. C., & Varoquaux, G. 2011, *CSE*, **13**, 22