Determining the parameters of extrasolar planets with machine learning algorithms

Laura Busak[1], Nikola Jurković[2], Luka Penjin[3], Philipp Matern[4], Boris Majić[5]

[1]V. Gymnasium, Zagreb, Croatia
[2]Gymnasium Metković, Metković, Croatia
[3]XV. Gymnasium, Zagreb, Croatia
[4]Hebel-Gymnasium, Schwetzingen, Germany
[5]University of Belgrade, Belgrade, Serbia

ABSTRACT

Extrasolar planets are planets located outside the solar system. There are different methods of finding exoplanet candidates and confirming their existence, but the most successful one is transit photometry. It uses data concerning the relative brightness of a star to find out whether a planet is transiting in front of its disk and determine some of the planet's properties, for example its orbital period. A huge amount of this data is obtained by telescopes and can be processed much more efficiently by machines than by humans [1]. This is why we developed a neural network that uses the transit method to get some information about the planet from the telescope-obtained data.

Keywords: transit photometry; exoplanets; machine learning

I. INTRODUCTION

There are many reasons why we are searching for extrasolar planets. Detecting exoplanets and learning more about them can, for instance, improve our understanding of planetary systems and their formation. Finding exoplanets in the habitable zone is also the first step to finding extraterrestrial life.

As we have eight planets in our solar system, scientists have suspected the existence of extrasolar planets for a long time but had not been able to make any confirmed detections until the end of twentieth century. Early extrasolar planetary discoveries were based on radial velocity measurements, which use the shift in the spectrum of a star caused by the Doppler effect to measure changes in the star's velocity due to the gravitational pull of a planet, but today we discover the majority using transit photometry. Nowadays, space telescopes reduce noise caused by distortions of light when passing through the atmosphere, which is one reason why we have been able to detect and confirm 3360 extrasolar planets so far, 2740 of which using the transit method [2].

Since a planet transiting in front of a star periodically causes a small change in its brightness, the dip in brightness can be represented with a graph known as the light curve.

Considering the exponential increase of computing power in the last few decades and since there was a lot of progress in artificial intelligence, it only makes sense to use computers and AI to process the telescope data and discover exoplanets.

In this paper, we explain how we first simulated a planetary system and a transit to get the training data for a neural network that can determine the ratio of the planet's and the star's size and the distance between them from the brightness-data of the star.

I. METHODOLOGY

Our goal was to make a functioning machine learning program, or more precisely, a neural network and train it to analyse light curves. In order to work properly and accurately, machine learning programs require a lot of data to train. We had to make a program that can simulate transits so we could feed generated data into our neural network and after that use it to get parameters of planets using real transits.

First step to creating our simulation program was to simulate a simple orbit of a planet around a star. We made the orbits, stars and planets perfectly circular to keep it simple. Also to make it simple, we didn't make any planet inclinations and just made the planet pass over the exact center of the star. Using that orbit, first we calculated how much of the star the planet is covering. Since the systems are very far from us, we didn't have to take the distance between the planet and the star into account. We started making graphs of transits by assuming the stars were uniformly bright across surface of their disk. We made graphs out of Numpy arrays in which we stored the numbers that represented the amount of light the sun is outputting at any given moment. We did the calculations as if they are circles in a 2D plane and used the formula for circle intersection when it was a partial intersection. When it was a full intersection, we subtracted the area of the planet from the area of the star.

$$A = \pi \times R^2 - \pi \times r^2 \tag{1}$$

$$A = r^2 \times cos^{-1}\left(\frac{d^2+r^2-R^2}{2*d*r}\right) + R^2 * cos^{-1}\left(\frac{d^2+R^2-r^2}{2*d*R}\right) - 0.5 * b \tag{2}$$

$$b = \sqrt{(-d+r+R)(d+r-R)(d-r+R)(d+r+R)} \tag{3}$$

where *r* and *R* represent the radius of the planet and the star respectively, *d* is the distance between the centers and *A* is the intersecting area.

Then we had to take limb darkening into account. Limb darkening is a phenomenon that explains why and how stars are not uniformly bright. The farther away a point on the star is from the center, the darker it is. Since we didn't understand the mathematics needed, we split the area of the star into a finite amount of rings as an approximation (Fig. 1.). We split it into rings because the effect of limb darkening is the same for every point that is the same distance from the center. We used the quadratic limb darkening law [3] to calculate the coefficient that describes how surface brightness changes as a function of distance away from the center of the disk.

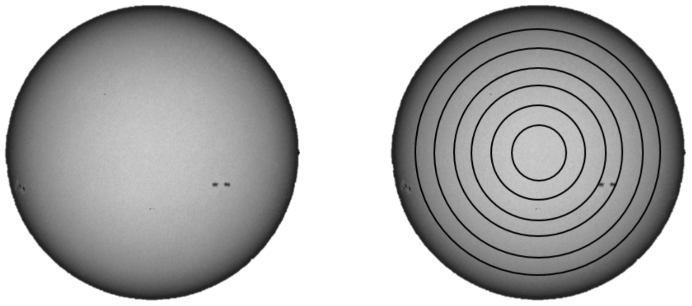
Figure 1. Depiction of the rings used to approximate the

First, we calculated the maximum amount of light the star outputs when there is no intersection. To approximate the amount of light the planet is blocking when a transit is in progress, we calculated the darkening coefficient for the middle point of the intersecting area. We assumed that coefficient applied to the whole area in transit and used it to calculate the amount of light blocked. After that we subtracted the maximum amount of light by the amount the planet is blocking. Now we could simulate a correct transit graph. Due to the fact that real data is afflicted by intrinsic noise from the telescope or natural variation in the star's brightness, we added random noise using the normal distribution, so that smaller errors are much more common than bigger errors.

After that, we simulated thousands of different random orbits for the neural network. We made an interval of numbers for each of the parameters for the orbit. We generated random numbers that were in that interval. We simulated 50 000 different transits and used 70 percent of the data for training and the other 30 percent for testing.

For the neural network we used Theano and Keras [4]. Theano is a library that allows us to make a neural network and Keras is a library that simplifies the creation of neural networks. Before starting the neural network, all data has to be normalized, which means that the overall average is 0, meaning that most of the numbers are in the interval between -1 and 1. We had to normalize our data since we used a pre-built neural network which was optimized for that kind of data, but in general, there are many advantages to working with normalized data.

After that we had to decide on the number of layers and nodes. We stuck with two output nodes for only two out of four parameters for now. We didn't do more since it's harder to make a neural network accurate the more output layers there are. We also made a dropout layer to prevent overfitting, which means that the neural network gets accustomed to the training data and does not perform very well with different data. Neural network algorithms are optimized for numbers of nodes that are powers of two, but there is no predefined approach to finding the right number of hidden nodes that works well, which is why we used trial and error to find that 128 nodes in the hidden layer works the best for us.

I. RESULTS

Using a Python program, we simulated light curves for a planet of a set size (Fig. 2.). It first produced graph shown in Fig. 2. a), with a flat line where real life examples show a curve. Our program assumed the star to be uniform across its entire surface, which is why the transiting planet would block the same amount of light as long as its whole area was in front of the star. This would not suffice, which is why we took limb darkening, a phenomenon further explained in Methodology, into account. The modified program returned a more satisfactory graph displayed in Fig. 2. b). Fig. 2. c) shows the graph with added noise, as explained in Methodology. These light curves proved the accuracy of our simulated data.

We fed the newly generated data into a neural network described in Methodology and compared the parameters it returned to their predicted values. This gave us the error distribution, which allowed us to test the neural network's accuracy. Fig. 3. shows three different histograms, each of which plotted the error distribution for a particular window of system parameters (Table 1.) used while generating data for the network to use. The results can be seen in Table 2. The parameters we used to describe the simulated systems were planet to star ratios, the ratios between the planet's and the star's radius, and the semimajor axis, the planet's distance from the star.

The error distribution in Fig. 3. c) and a median error of 2.86% and 9.62% for the planet-star ratios and axes respectively (Table 2.) point to a decidedly higher level of accuracy when compared to Figures 3. a) and b). This is related to a smaller window of greater planet to star ratios and lesser semimajor axes.

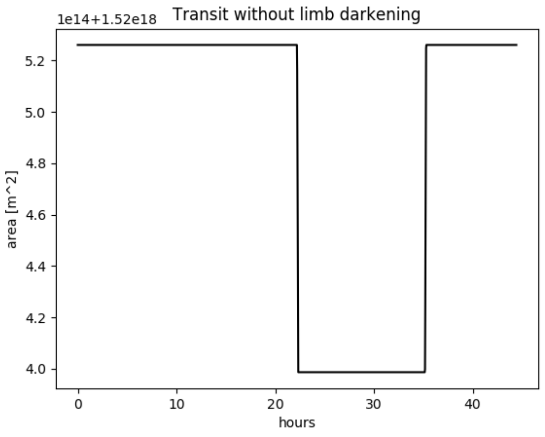

Figure 2. a). Transit graph without adjustment for limb darkening.

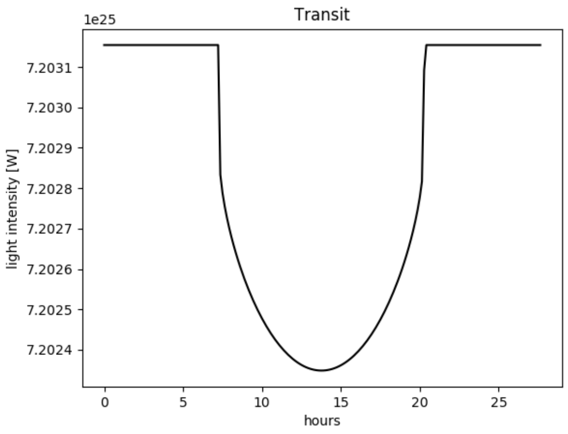

Figure 2. b). Transit graph with adjustment for limb darkening.

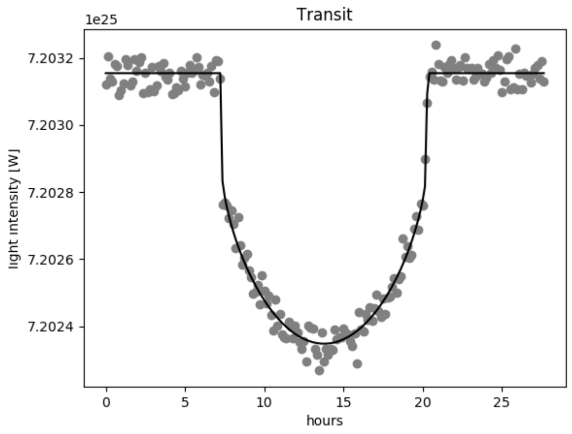

Figure 2. c). Transit curve with adjustment for limb darkening and simulated noise.

Figure 2. Simulated change in light output by a star during planetary transit.

Table 1. Parameters used for each of the three training sets.

Set		Planetary radius	Stellar radius	Semimajor axis
Narrow				
	Minimum	2.00×10^7	1.00×10^8	4.00×10^9
	Maximum	4.00×10^7	2.50×10^8	1.00×10^{10}
Normal				
	Minimum	1.00×10^7	1.00×10^8	4.00×10^9
	Maximum	6.00×10^7	5.00×10^8	4.00×10^{10}
Wide				
	Minimum	4.00×10^6	1.00×10^8	2.00×10^9
	Maximum	8.00×10^7	1.00×10^9	1.00×10^{11}

Table 2. Error distributions for three different training sets. The parameters used for each set correspond to those in Table 1.

	Normal		Wider		Narrow	
Errors	Planet-star radius ratio	Axes	Planet-star radius ratio	Axes	Planet-star radius ratio	Axes
Average	21.26%	34.24%	90.05%	53.29%	4.65%	12.61%
Median	8.11%	22.24%	25.50%	28.79%	2.86%	9.62%

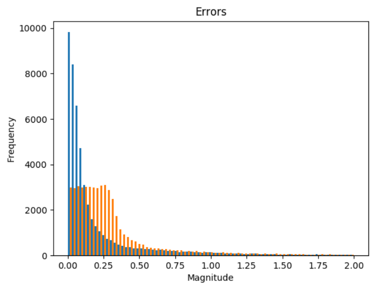

Figure 3. a). Normal distribution.

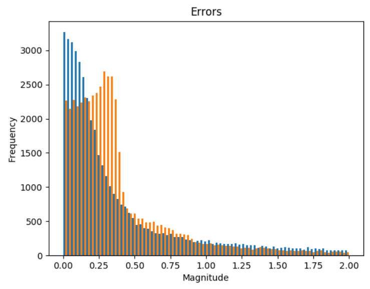

Figure 3. b). Wide distribution.

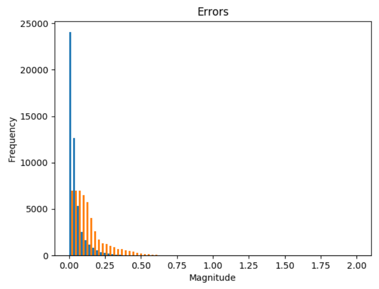

Figure 3. c). Narrow distribution.

Figure 3. Error distributions for three training sets. Showing errors for planet to star ratios and semimajor axes.

II. DISCUSSION

The different distributions of errors point to a correlation between the accuracy of the result and the parameters of the system. The simulated data seemed less sensitive to random noise when the dips in brightness were greater, which was directly related to greater planet to star ratios and lesser semimajor axes. Higher sensitivity to random noise leads to less accurate predictions of planetary parameters, meaning the neural network is currently better at predicting the properties of an extrasolar planetary system with greater planet to star ratios and lesser semimajor axes.

What we have done so far leaves a lot of areas unfinished and open to improvement. While we have set up a strong base for a successful neural network, there is still a lot of work to be done to improve its accuracy and overall functionality. Furthermore, all the orbits we simulated were circular and had an inclination of zero degrees. Using elliptical orbits would further improve accuracy and simulating orbits with different inclinations would increase sample size and real life usability a lot because inclination is almost never zero in real life. The usage of calculus instead of our ring model to implement limb darkening would increase accuracy as well. In addition, our neural network needs to be tested and further trained using real instead of simulated data.

III. ACKNOWLEDGMENTS

We would like to thank Sebastijan Dumančić and Nikolina Šoštarić, the organizers of Summer School of Science, and Požega Gymnasium for allowing us to work.

IV. REFERENCES

[1] Zhang, Y. & Zhao, Y., (2015). Astronomy in the Big Data Era. Data Science Journal. 14, p.11.
[2] http://exoplanet.eu/catalog/ ; 31.08.2017 / 1:13 pm
[3] Sing, D. K. “Stellar limb-darkening coefficients for CoRot and Kepler”, Astronomy and Astrophysics, Volume 510, (200); arXiv:0912.2274 [astro-ph.EP]
[4] F. Chollet, 2015, https://github.com/fchollet/keras