Udacity Machine Learning Nanodegree

Capstone Proposal: Exoplanet Hunting in Deep Space

Graciano Patino

September 2018

Domain background

The domain for this proposal is the field of astrophysics. In particular, the domain is about the fascinating topic of the search for new Earths, also called exoplanets. An exoplanet is a planet outside our Sun's solar system. The first evidence of an exoplanet was noted as early as 1917, but was not recognized as such. However, the first scientific detection of an exoplanet was in 1988, although it was not confirmed to be an exoplanet until later in 2012. The first confirmed detection occurred in 1992. As of 1 September 2018, there are 3,823 confirmed planets in 2,860 systems, with 632 systems having more than one planet. (Source: Reference 1).

In the past few decades, scientists now have at their disposal the more appropriate tools for detecting these planets. The Kepler space observatory provides images that can be analyzed on the search for exoplanets. In the case of this proposal, we can argue that machine learning provides a number of analytical tools to help on the detection of these exoplanets.

Problem Statement

The problem presented in this proposal was selected from the set of public datasets available at the kaggle website (reference 2). Additional information is provided in the following Github (reference 3.

The mission as stated in the Github (above) is to build a classification algorithm for identifying if a particular time series input includes an exoplanet or not. Basically, it is a binary classifier.

From the Overview tab in the mentioned kaggle dataset (above), this is high level description of the science involved on finding the exoplanets:

- The dataset describes the change in flux (light intensity) of several thousand stars. Each star has a binary label of 2 or 1. 2 indicated that that the star is confirmed to have at least one exoplanet in orbit; some observations are in fact multi-planet systems.
- As you can imagine, planets themselves do not emit light, but the stars that they orbit do. If said star is watched over several months or years, there may be a regular 'dimming' of the flux (the light intensity). This is evidence that there may be an orbiting body around the star; such a star could be considered to be a 'candidate' system. Further study of our candidate system, for example by a satellite that captures light at a different wavelength, could solidify the belief that the candidate can in fact be 'confirmed'.

Datasets and inputs

As described in Kaggle for this proposal (reference 2). The following training and testing sets are provided in two comma separated values file (CSV). The data is clean according to source.

Training set (File: exoTrain.csv):

- 5087 rows or observations.
- 3198 columns or features.
- Column 1 is the label vector. Columns 2 - 3198 are the flux values over time.
- 37 confirmed exoplanet-stars and 5050 non-exoplanet-stars.

Testing set (File: exoTest.csv):

- 570 rows or observations.
- 3198 columns or features.
- Column 1 is the label vector. Columns 2 - 3198 are the flux values over time.
- 5 confirmed exoplanet-stars and 565 non-exoplanet-stars.

It is noted in Github, that the datasets are not normalized.

Solution statement

The mission as stated in the Github (reference 3) is to build a classification algorithm for identifying if a particular time series input includes an exoplanet or not. It also mentions that a number of methods were tested: 1-D CNN in Torch7, XGBoost in R and PCA in Python. However, none of these methods provided strong results according to the kaggle and Github references (mentioned above).

For this project, I would evaluate deep learning algorithms. Per paper in the paragraph (below), these algorithms appear to provide better results compared to the ones already tried as mentioned above.

1) Initially I would evaluate 1-D CNN using Keras instead of Torch7.
2) Based on reference paper, I would try adding different number of layers and filters in combination with other CNN parameters. Details would be included in project report.
3) The output of the CNNs would be the input to one or more dense layers.
4) Performance of each model to be measured as per evaluation metrics section.
5) Per kaggle source the test set is confirmed to have 5 exoplanets. This will also be useful on checking performance of algorithms. If an algorithm is unable to identify exoplanets on then testing set, then model might not be good.

Please that the list above of models considered is not meant to be exhaustive for all possible scenarios in deep learning algorithms. It might be the case that other deep learning algorithms might be considered later should the ones proposed (above) fail in identifying any exoplanet as expected.

Check reference 4 paper for similar problem: IDENTIFYING EXOPLANETS WITH DEEP LEARNING: A FIVE PLANET RESONANT CHAIN AROUND KEPLER-80 AND AN EIGHTH PLANET AROUND KEPLER-90.

Benchmark model

According to Github (reference 4) a number methods were tested (1-D CNN in Torch7, XGBoost in R and PCA in Python). In the mentioned Github, none of these methods had strong results.

In the XGBoost example, the algorithm was unable to identify any of the 5 exoplanets that are included in the set (according to kaggle source). All samples were classified as non-exoplanets.

For reference, I would evaluate the 1-D CNN using Keras instead of Torch 7. In theory, this would have very similar results as with Torch 7. The expectation is that a single layer CNN would not perform very well on successfully classifying exoplanets when present in the testing set.

Evaluation metrics

For assessing our model's performance, I will use metrics like:

- Receiver Operating Characteristic (ROC) metric to evaluate classifier output quality. (Scikit)
- Area under the Receiver Operating Characteristic Curve (ROC AUC) from prediction scores.
- Precision: the fraction of signals classified as planets that are true planets (also known as reliability, per reference 4).
- Recall: the fraction of true planets that are classified as planets (per reference 4).
- Accuracy: the fraction of correct classifications.

Please note that some or all of the metrics above will be used for assessing model performance. However, this list is not meant to be exhaustive at the time of writing this presentation.

Project design

The workflow for solving this problem would have the following order:

- Exploring the datasets (check dimensions of data, labels, etc.)
- Data preprocessing:
    - The datasets provided by kaggle (reference 2) are supposed to be clean.
    - It is noted that the data is not normalized.
    - Github (reference 3) mentions that techniques like data augmentation could help as we are dealing with time series. Perhaps systematically shifting rows and adding noise could generate additional realistic (albeit synthetic) trends. This might be explored depending on results from the different algorithms tested for solving the problem.
- Evaluate machine learning algorithms: This involves building the models and selecting best model by using evaluation metrics and comparing to benchmark model(s).
- Model tuning to optimize results: This involves using evaluating performance of the model and fine tuning hyper-parameters until a satisfactory model is identified. This is by using evaluation metrics to evaluate performance.
- Final conclusions.

Please note that proposed workflow is provided at a high level. Additional detail would be provided in the project report.

References

1) Wikipedia on exoplanets: https://en.wikipedia.org/wiki/Exoplanet
2) Exoplanet search dataset in kaggle: https://www.kaggle.com/keplersmachines/kepler-labelled-time-series-data/home
3) Github associated to (2): https://github.com/winterdelta/KeplerAI.
4) Paper on : IDENTIFYING EXOPLANETS WITH DEEP LEARNING: A FIVE PLANET RESONANT CHAIN AROUND KEPLER-80 AND AN EIGHTH PLANET AROUND KEPLER-90 (https://www.cfa.harvard.edu/~avanderb/kepler90i.pdf