Formation, Detection, and Characterization of Extrasolar Habitable Planets
Proceedings IAU Symposium No. 293, 2012
N. Haghighipour, ed.

doi:10.1017/S1743921313012611

Auto-Vetting Transiting Planet Candidates Identified by the Kepler Pipeline

Jon M. Jenkins[1], Sean McCauliff[2], Christopher Burke[1], Shawn Seader[1], Joseph Twicken[1], Todd Klaus[2], Dwight Sanderfer[3], Ashok Srivastava[3] and Michael R. Haas[3]

[1]SETI Institute/NASA Ames Research Center, M/S 244-30, Moffett Field, CA 94035
email: jon.jenkins@nasa.gov

[2]Orbital Sciences Corp./NASA Ames Research Center, Moffett Field, CA 94035
[3]NASA Ames Research Center, Moffett Field, CA 94035

Abstract. The *Kepler* Mission simultaneously measures the brightness of more than 150,000 stars every 29.4 minutes primarily for the purpose of transit photometry. Over the course of its 3.5-year primary mission *Kepler* has observed over 190,000 distinct stars, announcing 2,321 planet candidates, 2,165 eclipsing binaries, and 105 confirmed planets. As *Kepler* moves into its 4-year extended mission, the total number of transit-like features identified in the light curves has increased to as many as ∼18,000. This number of signals has become intractable for human beings to inspect by eye in a thorough and timely fashion. To mitigate this problem we are developing machine learning approaches to perform the task of reviewing the diagnostics for each transit signal candidate to establish a preliminary list of planetary candidates ranked from most credible to least credible. Our preliminary results indicate that random forests can classify potential transiting planet signatures with an accuracy of more than 98.6% as measured by the area under a receiver-operating curve.

Keywords. methods: data analysis, methods: statistical, instrumentation: photometers, (stars:) planetary systems, catalogs

1. Introduction

The *Kepler* Mission has generated three exoplanet candidate catalogs in the first 3.5 years of operation: Borucki *et al.* (2011a), Borucki *et al.* (2011b), and Batalha *et al.* (2012). The process of compiling the list of candidates from the science pipeline output is a very intensive manual effort, as a run across 30 months of data will typically identify $\gtrsim$18,000 transit-like signatures, or threshold-crossing events (TCEs). The vast majority of these signatures are due to instrumental artifacts mimicing transits from the standpoint of a linear transit detector. While the pipeline also provides a suite of valuable diagnostics to reduce the initial list down to a more manageable 5,000 or so TCEs, these filters can overlook valid planetary candidates. Moreover, the large number of diagnostics (∼100) makes it difficult for humans to examine all the information available to appropriately rank the TCEs.

The vetting process has evolved over time, and consists of two steps: 1) The TCEs are "triaged" based largely on an examination of the light curves and the transit fit parameters to construct an initial list of *Kepler* Objects of Interest (KOIs). 2) The KOIs are then vetted in detail considering other diagnostics such as difference image centroid information and a more thorough inspection of the diagnostics produced by the pipeline to generate a disposition for each KOI (planet candidate, astrophysical false positive, other instrumental or stellar variability effect, or uncertain; Batalha *et al.* 2012). The effort required to complete the triage and vetting steps have typically occupied several

months of effort by many individuals in the *Kepler* Science Office and Science Team in order to winnow the ∼18,000 transit-like signatures down to the final planetary candidate list published in each new catalog. As the science pipeline has been improved, the number of identified transit-like signatures has increased, compounding the effort.

To mitigate this growing problem, we are pursuing machine learning approaches to reduce the time required by humans to separate the obvious chaff from the wheat. Machine learning is a branch of computer science and statistics that has generated a number of algorithms that are useful for classifying and clustering data, usually in high dimensional spaces. The ideal algorithm would generate a list of candidates that closely approximates those generated by human review of the pipeline diagnostics, and would rank each transit-like signature from most credible to least credible, thereby allowing the humans to focus on the most interesting cases. Machine learning approaches can also help identify the most important metrics and diagnostics with respect to separating signatures of transiting planets and eclipsing binaries from instrument-induced features.

In this paper we give a brief overview of the transit search pipeline and discuss those instrumental signatures most responsible for false positives. We then describe the machine learning approach under study for both triage and vetting, and present preliminary results with an "auto-triage" algorithm. We conclude with thoughts on the future effort.

2. The *Kepler* Transit Search Pipeline

The systematic error-corrected *Kepler* light curves are subjected to an adaptive, wavelet-based matched filter to identify transit-like features; sets of three or more periodically spaced negative rectangular pulses from 1.5 to 15 hours in duration, and with periods from 0.5 days to half the length of the data set (Jenkins 2002, Jenkins, Chandrasekarana, McCauliff, *et al.* 2010). Transit-like features that appear to be statistically significant (i.e., SNR $> 7.1\ \sigma$) and that pass the internal vetting inside of TPS are called TCEs. Light curves with TCEs are subjected to a suite of diagnostic tests by the Data Validation (DV) pipeline in order to build or break confidence in the TCE as a planetary signature (Wu *et al.* 2010). DV also fits a limb-darkened light curve to each potential planetary signature and then removes the signature of the current TCE and calls TPS on the residual light curve in order to detect additional planetary signatures.

Despite both the internal vetting that occurs in TPS, and the more sophisticated modeling and testing performed by DV, the majority of the TCEs that result from a TPS/DV run are false positives caused by transients in the data. For example, imperfectly corrected thermal transients (which modify the optical focus and plate scale) or occasional negative step discontinuities caused by radiation damage to the CCDs can result in TCEs (Tenenbaum *et al.* 2012). A new class of false positives with periods of ∼1 year is emerging as the search extends through several years of data. These are caused by transient image artifacts due to flaws in the electronics (Van Cleve & Caldwell 2009). The repeatability of the thermal profile and pointing helps exacerbate the situation as the artifacts often repeat at the same place in the orbit. Such false positives are more likely for candidates featuring few transits. While the DV diagnostics can be used to easily discard most of these false positives, it remains an intensive manual effort as DV only presents the information; DV does not act upon the information to construct a list of planetary candidates.

At this point in the mission, continuing to rely solely on manual vetting of the DV output to construct the KOI lists is simply no longer tenable. Therefore we have initiated development of an automated method of vetting the DV output to construct a preliminary list of planetary candidates. Section 3 describes this effort.

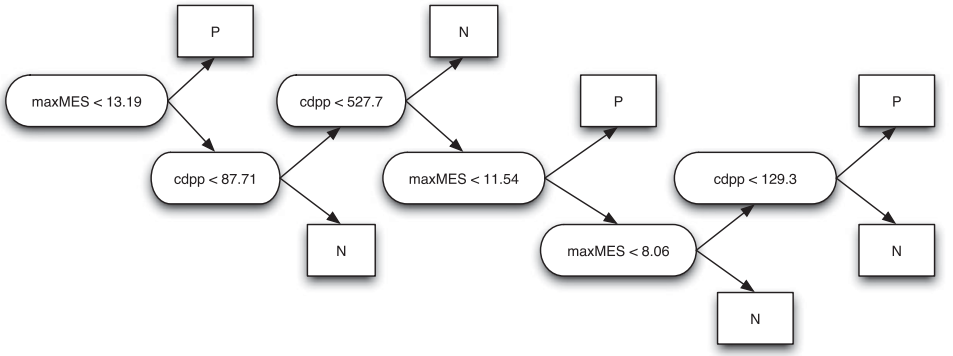

Figure 1. Example decision tree for deciding between "planet" (P) and "non-planet" (N) using only the star's CDPP and the signal's maximum multiple event statistic ($maxMES$).

3. Decision Trees and Random Forests

Classification algorithms come in pairs. A training algorithm builds the classifier (the model), Ω, from the training data set $\mathcal{D}$. $\mathcal{D}$ is composed of observations $\mathbf{X} = \{\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_M\}$ and class assignments Y from the set $C = \{C_1, ..., C_R\}$. A prediction algorithm is a function $Y = h(\mathbf{X}, \Omega)$ that uses the model to assign a C_i to an unclassified observation $\mathbf{x}_i \in \mathbf{X}$ with attributes from the set $A = \{A_1, ..., A_N\}$. For example, an attribute for a planet classification problem might be "orbital period". Attributes can be continuous, discrete or categorical. Classification prediction algorithms can be viewed as a form of regression where the predicted variable is categorical in nature.

A decision tree is much like the "20-questions" game; each question can be viewed as a node in the tree as in Figure 1. Branches in the tree indicate "yes" or "no" answers to the question posed in the node (multi-way branches are also possible, but can be represented using binary decision trees). Branching proceeds until the no other questions can be asked, this results in a leaf node; that is a classification. Decision trees have the advantages of being interpretable models of classification: one can just look at the decision tree and answer the questions to arrive at a classification. Another, perhaps more important, advantage is that decision trees do not assume any particular distribution of the data and do not require that attributes have any particular distance relation. Decision trees produce rectangular decision boundaries perpendicular to attribute axes; in higher dimensional spaces the decision boundary is composed of hyper-rectangles. With a sufficiently large tree any decision boundary can be approximated.

Classification and Regression Trees (CART) is a decision tree learning algorithm invented by Breiman, Friedman, & Stone (1984). Each node in a tree produced by CART asks the question is $x_i \leqslant x_{is}$? CART produces this inequality by splitting $\mathcal{D}$ into subsets $\mathcal{D}_k$ with each question asked. Each split is made preferring the simplest possible decision tree with the fewest nodes. To this end CART seeks an attribute that will split $\mathcal{D}_k$ into two subsets each with the greatest purity with respect to Y. CART actually minimizes the impurity of a node rather than maximizing the purity. The impurity of node k in the tree is represented as $I_G(\mathcal{D}_k)$. $I_G = 0$ when $\mathcal{D}_k$ all have the same class. I_G should be a large number when the mixture of classes reaching the node are nearly equal in frequency. CART uses the Gini impurity function defined as

$$I_G = 1 - \sum_r \hat{P}^2(C_r|\mathcal{D}_k), \tag{3.1}$$

where $\hat{P}$ is the estimated probability of its argument. When creating a split the algorithm chooses the x_{is} of attribute A_i that minimizes

$$\Delta I_G = I_G(\mathcal{D}_k) - \Delta I_G(\mathcal{D}_{k,true}) - \Delta I_G(\mathcal{D}_{k,false}), \tag{3.2}$$

of child nodes $\mathcal{D}_{true,false}$. When the number of classes reaching a node has fallen to one, or some threshold on the size of the subset of $\mathcal{D}_k$ is reached, a leaf node which assigns a class to an attribute vector, $\mathbf{x}$, is generated.

Decision trees can be sensitive to noise, in the form of either outlier values in $\mathbf{X}$ or mistaken classifications in Y (Breiman *et al.* 1984). Some implementations use heuristics to prune the decision tree to reduce the over-fitting error. Random forests introduced by Breiman (2001) remedy this problem by creating an ensemble of decision trees and averaging their classifications.

The random forest training algorithm builds a set of classifiers $\{h(\mathbf{X}, \Omega_j)\}$ where Ω_j is the j^{th} decision tree trained on random subset $\mathcal{D}_j$. $\mathcal{D}_j$ is a bag; a random subset with replacement. Bag creation can be weighed by the prior probabilities of C_r so that $\mathbf{x}_i$ is chosen with respect to its y_i. When the number of bags, B, becomes very large the misclassification error will converge. A typical value of B is 1000.

Random forests have a theoretical upper bound to their misclassification error that depends on the correlation of their component decision trees; higher correlation yields higher error bounds. To reduce correlation between base classifiers the training algorithm introduces randomness into the creation of each Ω_j. At each node in Ω_j a random subset of A is chosen. The split minimizing Eq. 3.2 is made, but only among the randomly chosen subset of A. The number of A to select, m_{try} is an algorithm parameter; it is typically $\lceil\sqrt{N}\rceil$. The training algorithm is not particularly sensitive to the value of m_{try}.

4. Labeled Data Set

We are interested in generating a classifier that can distinguish between planet candidates and instrumental, or systematic noise sources that mimic planet transit signals. One source of planet candidates comes from the third KOI catalog (Batalha *et al.* 2012). The KOI catalog is an exoplanet catalog composed of confirmed planets, planet candidates and astrophysical false positives. We drop from consideration systems with eclipsing binaries, such as Kepler-16b (Doyle *et al.*2011), a circumbinary planet. The second *Kepler* eclipsing binary catalog from Slawson *et al.* (2011) is used for this purpose. While astrophysical false positives are an important case to distinguish from actual candidates the training data set does not make any efforts to exclude astrophysical false positives. For example, background eclipsing binaries causing transit-depth signatures on affected target stars remain in the training data set. The KOI catalog also includes some planet candidates identified from only a single transit. Since TPS is limited to detecting periodic signals, such candidates are excluded. Additional planet candidates were added by visually inspecting DV reports (Twicken, *et al.* 2012). A DV report contains the exoplanet ephemeris, light curves, and the results of several tests presented in a human readable format. Each DV report was reviewed by two scientists. If either of them thought the signal described by the report indicated a potential new transiting planet then this was added to the KOI list. The DV reports will become available to the general public sometime in late 2012. TCEs not marked as planet candidates by either human vetter are assigned the class of "noise" in the training data set.

The run of the *Kepler* pipeline used in this paper has a total of 16,955 signals from approximately 190,000 stars over the first quarter to the tenth quarter of *Kepler* data.

All of these signals passed TPS's internal statistical tests (Tenenbaum *et al.* 2012). Some of the these signals represent transiting planet candidates in multiple planet systems. 3338 training examples for class "planet candidate" match the ephemerides produced by the *Kepler* pipeline. 1085 training examples match the examples of class "noise". Each training example has 31 attributes. The actual prior probability of a planet around a star is not known and is an active area of research. However, we believe the relative frequency of planet examples greatly overestimates the actual frequency of planets detectable by *Kepler*. So the prior probability of 5/8 is used to weight occurrence of noise examples used to construct the bags of the random forest.

5. Classifier

5.1. *Classifier Accuracy*

A random subset of 70% of the labeled data was used for training. The remaining 30% was used to evaluate classifier performance. The default value of m_{try} was used. A search over different values of B was performed between 50 and 5000 trees. After about 1000 trees there is less than 1% change in the out of bag error; that is the mean error computed on each tree in the forest on the data not in its bag of training examples.

After the random forest is trained a prediction is made on each signal in the test set. During the prediction each tree in the forest votes for the item in the test set. This vote can either be for a planet candidate or systematic instrumental noise. The number of votes for each signal can be used to order the predictions from most number of votes for planet candidate to fewest votes for planet candidate. Knowing the actual classification of the signal in the test set allows the construction of Figure 2 which shows the trade off between the true positive rate vs. the false alarm rate. This decision threshold is completely controlled by the number of votes for each class.

Figure 2 shows the receiver-operating characteristic curve (ROC) for the classifier, which is a graph of the true positive classification rate (y-axis) as a function of the false positive classification rate (x-axis). A random classifier would have a curve that is close to the $y = x$ line, since at any point the rate of false positives would approximately equal the rate of true positives. The higher the curve is away from the $y = x$ line, the better the classifier. Our classifier's true positive rate is well above 90% at a false alarm rate of 1%, indicating it is outperforming a random model significantly.

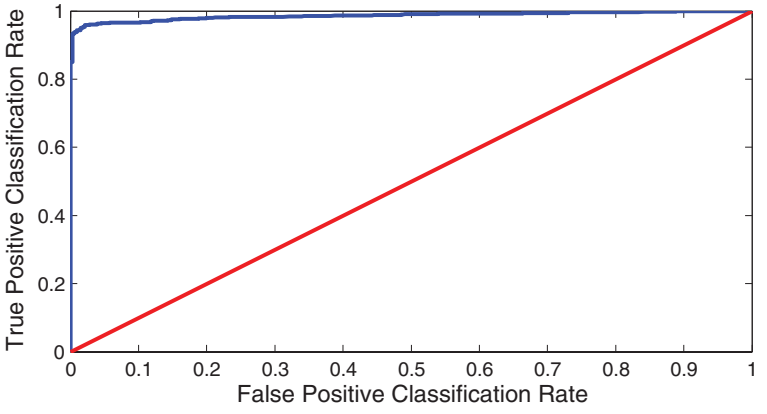

Figure 2. ROC curve for the random forest trained for auto-triage.

5.2. *Attribute Importance*

The random forest can be used to estimate the importance of each attribute during training. For each tree's out of bag data each attribute is shuffled in turn. The tree then makes a prediction for each out of bag signal. The mean increase in error for an attribute over all trees is then its importance. For the auto-triage results presented here, we used a subset of the DV diagnostics that would be available from TPS, such as the multiple event statistic, *mes*, the ratio of *mes* to the max or min single event statistic, *ses*, the depth of the transit, the number of transits, the epoch of the transit sequence, a transit shape static, and the degree to which the *ses* time series are correlated. A full description of the metrics used in the auto-triaged is beyond the scope of this introductory paper, and will be reported in a future journal-length article.

6. Conclusions

We have described the application of the random forest algorithm to the problem of classifying the output of the *Kepler* planet search pipeline so that we can automate the construction of a preliminary list of planet candidates from the pipeline output. The random forest algorithm has several advantages: 1) The number of trees voting for the class assigned to each potential planetary signature can be used as a means to rank the set of TCEs from most credible to least credible, thus allowing us to allocate the human resources to the most productive light curves, 2) The importance of each diagnostic (attribute) can be determined objectively by randomizing each diagnostic then examining the degradation in performance of the resulting classifier, 3) The random forest can be used to investigate the performance and bias of the human vetting process. Our future work will focus on improving the quality of training set to include more examples of instrumental, systematic noise. We are also working on simulating transits within the *Kepler* pipeline for the purposes of estimating the completeness of the *Kepler* Mission and provide ground truth for the purpose of evaluating the classifier.

Acknowledgements

Kepler was selected as the 10th mission of the Discovery Program. Funding for this mission is provided by NASA's Science Mission Directorate.

References

Batalha, N. M., Rowe, J. F., Bryson, S. T., *et al.*, *ApJS*, 204, article id. 24

Breiman, L., Friedman, J. ,Olshen, R., & Stone, C. 1984 *Classification and Regression Trees.* Boca Raton, FL: CRC Press.

Breiman, L. 2001 *Machine Learning*, 45, 1

Borucki, W. J., Koch, D. G., Basri, G., *et al.* 2011a, *ApJ* 728, 117

Borucki, W. J., Koch, D. G., Basri, G., *et al.* 2011b, *ApJ* 736, 19

Doyle, L. R. Carter, J. A., Fabrycky, D. C., *et al.* 2011 *Science* 333, 6049

Jenkins, J. M. 2002, *ApJ* 575, 493

Jenkins, J. M., Hema Chandrasekarana, H., McCauliff, S. D., *et al.* 2010, *Proc. SPIE* 7740.

Slawson, R. W., Prša, A., Welsh, W. F., *et al.* 2011, *AJ*, 142, 160

Tenenbaum, P., Christiansen, Jessie L., Jenkins, J. M., *et al.* 2012, *ApJS* 199, 24

Twicken, J. D., Wu, H., Wohler, B., *et al.* 2012, *AAS Meeting* 220, abstract #330.05.

Van Cleve, J. & D. A. Caldwell, 2009, *Kepler Instrument Handbook*, KSCI 19033-001, (Moffett Field, CA: NASA Ames Research Center)

Wu, H., Twicken, J. D., Tenenbaum, P. *et al.* 2010, *Proc. SPIE*, 7740, 774019