Deep Learning Classification in Asteroseismology Using an Improved Neural Network: Results on 15000 Kepler Red Giants and Applications to K2 and TESS Data

Marc Hon,1* Dennis Stello,1,2,3 and Jie Yu2

1School of Physics, The University of New South Wales, Sydney NSW 2052, Australia
2Sydney Institute for Astronomy (SIfA), School of Physics, University of Sydney, NSW 2006, Australia
3Stellar Astrophysics Centre, Department of Physics and Astronomy, Aarhus University, Ny Munkegade 120, DK-8000 Aarhus C, Denmark

ABSTRACT

Deep learning in the form of 1D convolutional neural networks have previously been shown to be capable of efficiently classifying the evolutionary state of oscillating red giants into red giant branch stars and helium-core burning stars by recognizing visual features in their asteroseismic frequency spectra. We elaborate further on the deep learning method by developing an improved convolutional neural network classifier. To make our method useful for current and future space missions such as K2, TESS and PLATO, we train classifiers that are able to classify the evolutionary states of lower frequency resolution spectra expected from these missions. Additionally, we provide new classifications for 8633 Kepler red giants, out of which 426 have previously not been classified using asteroseismology. This brings the total to 14983 Kepler red giants classified with our new neural network. We also verify that our classifiers are remarkably robust to suboptimal data, including low signal-to-noise and incorrect training truth labels.

Key words: asteroseismology – methods: data analysis – stars: evolution – stars: oscillations – stars: statistics

1 INTRODUCTION

In constraining the stellar ages of red giants, it is important to determine their evolutionary state and distinguish between those that are currently burning hydrogen in a shell surrounding their inert helium core and those that have already commenced core-helium burning. The former are known as red giant branch (RGB) stars, while the latter are helium burning (HeB) or clump stars. Both types have overlapping populations in the colour-magnitude diagram, which makes it challenging to distinguish them using classic stellar surface properties such as luminosity and effective temperature. By observing the stochastically excited and damped, so-called solar-like, oscillation frequencies of red giants, asteroseismology effectively probes the interior stellar structure and has enabled classifications based on the observed dipole mode period spacings (e.g. Bedding et al. 2011; Stello et al. 2013), the asymptotic dipole mode period spacings (e.g. Bedding et al. 2011; Mosser et al. 2014; Vrard et al. 2016), and mixed mode frequency distributions (Elsworth et al. 2016).

In our previous work (Hon et al. 2017, hereafter H17), we introduced an asteroseismic deep learning method using 1D convolutional neural networks. The method classifies the oscillation spectra of red giants into RGB or HeB stars by recognizing the significant visual patterns shown in the frequency power spectra by the oscillation modes that are characteristic to the evolutionary state of the red giant. This automated method uses visual recognition to classify power spectra and is conceptually similar to computer vision applications such as facial recognition (e.g. Garcia & Delakis 2004), image detection and classification (e.g. Krizhevsky et al. 2012; Simonyan & Zisserman 2014), text categorization (e.g. Zhang et al. 2015), and biomedical image analyses (e.g. Cireşan et al. 2013; Prasoon et al. 2013).

With the advent of space missions K2 (Howell et al. 2014), TESS (Ricker et al. 2014), and PLATO (Rauer et al. 2014), the need for wide-scale red giant classification becomes ever more demanding, because the total amount of red giant data received surpasses the amount from the Kepler mission (Borucki et al. 2010) by orders of magnitude (Campante et al. 2016; Miglio et al. 2017; Stello et al. 2017). Moreover, the shorter time series from these newer missions make ‘classical’ asteroseismic methods for classification more challenging, because they do not have sufficient frequency resolution to resolve the complex frequency structure of the dipole modes. Hence, there exists a need to expand our method to missions like K2 and TESS. In doing so, we will be better prepared to analyse and interpret the incoming large volume of data efficiently and effectively.

© 2017 The Authors
In this paper, we expand on the deep learning method developed by H17 and introduce an improved version of the convolutional neural network classifier. We train versions of this new classifier on data lengths specific to Keplertual mission. (b, f) the length of 1 year is typical for the TESS continuing viewing zones. (c, g) the length of 82 days is typical for a K2 campaign. (d, h) and 27 days is representative of the shortest TESS observations.

**Figure 1.** Noise normalized power spectra (left) and corresponding folded spectra (right) of the RGB star KIC 3641504 for varying lengths of observation time. The frequency at maximum power, \( \nu_{\text{max}} \), the overtone frequency separation, \( \Delta \nu \), and the spherical degrees of the low degree acoustic modes are indicated. (a, e) The observation length of 4 years is representative of the full Kepler mission. (b, f) The length of 1 year is typical for the TESS continuing viewing zones. (c, g) The length of 82 days is typical for a K2 campaign. (d, h) and 27 days is representative of the shortest TESS observations.

2 DATA AND PERFORMANCE TESTING METHODS

Similar to H17, we use the folded spectrum as an image representation for the data in this study (Figure 1). The folded spectra are constructed from power spectra that are divided by their background noise, making them flat with a mean noise level of 1.0. To train and measure the performance of classifiers, we use a dataset comprised of 6015 Kepler red giants with classifications based on automated period spacing measurements from Vrard et al. (2016) and an additional 335 stars from the classification from Mosser et al. (2014) to a total of 6350 unique stars, with a ratio of RGB to HeB stars of approximately 1:2. We use the same 5000 stars as H17 as the training set, with the remaining 1350 stars as the test set. We assign RGB stars with the binary class 0 and HeB stars with class 1.

2.1 Shorter Time Series

To simulate the time series for K2 and TESS observations, we split the existing Kepler time series into multiple segments of shorter length. For K2, we used light curve segments of 82-day duration, while for TESS, we used 356-day, 82-day, and 27-day segments. Combined, the results from Kepler and our K2 and TESS simulations form a representative set for what one can expect from PLATO, depending on the observational strategy the mission will adopt. The effect of a shorter observation time on the quality of the power spectrum is illustrated in Figure 1. With shorter observations, the frequency resolution of the data decreases, the signal-to-noise for oscillation peaks decreases, and the visibility of resolvable peaks is reduced. While splitting the time series into smaller segments produces data of lower quality, it also produces multiple time series originating from the same star. This can potentially improve the performance of the classifier because the number of stars within the training set can be artificially boosted by having multiple ‘independent’ time series segments of a star within the dataset. These segments can be considered independent because they contain natural variations in noise and mode visibilities amongst themselves, such that their image representations are not too similar. We have verified that the classifiers do not produce the exact same predictions for multiple segments from the same star. The power spectra of multiple segments from the same star contain a sufficient level of variation such that they can be considered as different stars from another. The inclusion of these multiple segments in training allows the classifier to generalize better. We denote a dataset containing multiple segments from the same star as a degenerate set. We obtain degenerate training sets containing 14110, 66376, and...
193,534 stars for time series lengths of 356 days, 82 days, and 27 days, respectively.

In contrast, we denote a dataset without multiple segments of the same star as a non-degenerate set. In other words, all stars in a non-degenerate set have unique KIC identifiers within the set. We only use non-degenerate sets for the test set. Because the visual representation of the folded spectra can appear significantly altered by the changes in data quality across different lengths of time series, we train separate classifiers for each data length. Test sets of a particular length of time series will only be predicted on by a classifier that is trained on a degenerate training set of an identical length of time series. In this paper, we commonly use the term ‘data length’ for a particular dataset. This term references the length of the time series from which the dataset is created, not the dimensionality of the dataset.

2.2 $\Delta \nu$ Precision

As in H17, we obtain the $\Delta \nu$ values from the SYD pipeline (Huber et al. 2009) in order to generate the folded spectra. For the 4-year Kepler data we adopt the $\Delta \nu$ results from Yu et al. (in preparation). However, with shorter time series, we expect the $\Delta \nu$ values to be less precise. Because we want our datasets to be representative of K2 and TESS data not just in terms of frequency resolution but also in terms of $\Delta \nu$ precision, we simulate $\Delta \nu$ measurements for shorter data lengths. First, we measure the fractional deviation of 6090 stars’ 356-day, 82-day, and 27-day $\Delta \nu$ values from their 4-year $\Delta \nu$ values, which we show in Figure 2. While this set of stars is only a subset of the total number of classified red giants ($\approx 15,000$), they are approximately evenly distributed in evolutionary state population, and are only selected because they show a high level of continuity in each shorter time series segment. Next, to generate representative $\Delta \nu$ values for each 356-day, 82-day, and 27-day datasets, we let the $\Delta \nu$ value for each star be normally distributed about its 4-year $\Delta \nu$ value, with a standard deviation sampled from their corresponding uncertainty distributions in Figure 2. In order to obtain representative $\Delta \nu$ values for each star, we sample from each of their normal distributions. Using this method, multiple segments of the same star in a degenerate set (as discussed in Section 2.1) will have $\Delta \nu$ values slightly differing from one another. We now use these representative $\Delta \nu$ values to create our training and test sets for shorter time series.

2.3 Testing the Classifier’s Performance

We test the performance of a classifier using two general methods. The first is by testing on the test set, which provides an unbiased measure of performance because the classifier does not train on the test set. Another method is to partition the training data into $k$ separate sets or folds, then train on $k-1$ (training) folds and measure the performance on the remaining (validation) fold. This method is known as $k$-fold cross validation, and is useful because measuring the average metrics over $k$ independent sets can better account for the variance from predicting on a finite number of samples. However, the performance of the classifier reported by this method may be slightly underestimated as it does not learn from the full training set. A summary of the metrics we use to describe the classifier performance is as follows:

Accuracy: The number of correct predictions out of all predictions.

Precision(P): For a class, the ratio of correct predictions to all made predictions towards that same class. Here it is the classifier’s ability to not label a HeB star as an RGB star.

Recall(R): For a class, the ratio of correct predictions to all stars truly in that same class. Here it is the classifier’s ability to find all HeB stars.

F1 Score: The harmonic mean of precision and recall, defined by $2P \times R / (P + R)$, with 1 as a perfect score.

ROC AUC: Receiver Operating Characteristic’s Area Under Curve, which measures the classifier’s average performance across all possible score thresholds. Has a value of 1 for a perfect classifier.

Log Loss: Negative logarithm of prediction scores i.e. the cross entropy. Measures how well prediction scores are calibrated with an ideal value of 0 (see Equation 1).

Brier: Mean squared error between predicted probability and ground truth (see Equation 4).

3 THE NEW AND IMPROVED CLASSIFIER

We introduce a new and improved classifier structure, which takes the form of a combination of classifiers. In addition, we optimize the classifiers by performing probability calibration and determining the ideal probability threshold for separating RGB and HeB stars.

3.1 New Classifier Structure

In our previous work, we developed a deep learning classifier, which we denote here as the original model. We use the term
‘model’ to distinguish between the different types of classifier structures. Although the original model could predict the evolutionary phases of Kepler red giants with a high degree of accuracy, it had two limitations. The first was the confident HeB predictions for stars with high $\Delta \nu$ ($\geq 9 \mu$Hz), a range where HeB stars should not exist (e.g. Stello et al. 2013, their Figure 4b, and Mosser et al. 2014, their Figure 1). Secondly, the classifier predicted mainly very confident scores, with values very close to either 0 (RGB) or 1 (HeB) (H17, their Figures 6 and 7). While confident scores in general are not an issue, the lack of intermediate scores may suggest that the classifier is overfitting the data.

To address the first limitation, we develop a variation of the original model that predicts mainly RGB stars for $\Delta \nu \geq 9 \mu$Hz. We achieve this by providing $\Delta \nu$ as an additional input at the output layer of the classifier as illustrated in Figure 3. This altered version is denoted as the addition model. The effect of explicitly providing the network, or classifier, with values of $\Delta \nu$ is to provide a prior on the expected distribution of RGB and HeB stars, such that the classifier recognizes ranges of $\Delta \nu$ where certain populations do not occur.

We address the second limitation, the overconfidence of the original model, by creating a neural network structure that combines the predictions of both the original and the addition models, which we denote the combined model. By combining the output predictions of more than one neural network, we reduce the tendency to overfit the data (Hansen & Salamon 1990). We combine the predictions of the original model with the predictions of the addition model by a weighted average, where the weights are optimised depending on $\Delta \nu$. Using an optimal linear combination can, and in this case does, yield better results than simple averaging (Hashem et al. 1994). Thus, we manually tune the weights of each model by assigning greater value to the better performing model (Hashem et al. 1994). Thus, we manually tune the weights of each model by creating a neural network structure that combines the predictions of the original model, by creating a neural network structure that combines the predictions of both the original and the addition models, respectively. We reduce the frequency of having resulting severely incorrect predictions by H17, except here values of $\Delta \nu$ are added to the network (the bolded diagram branch). The numbers in brackets indicate the number of neurons in the layer.

Figure 3. Schematic of the structure of the altered deep learning classifier, denoted as the addition model. The structure is similar to the classifier structure by H17, except here values of $\Delta \nu$ are added to the network (the bolded diagram branch). The numbers in brackets indicate the number of neurons in the layer.

\[ E(y, \hat{y}) = -\frac{1}{m} \sum_{i=1}^{m} \left[y_i \log \hat{y}_i + (1 - y_i) \log (1 - \hat{y}_i) \right] \]

where $\hat{y}$ is a classification probability between 0 and 1, and $y$ is either 0 or 1. The lower the value of the log loss, the better the performance of the classifier in classifying stars. Reporting the log losses allows us to examine the performance of the classifier in terms of their probabilities. This is a more informative metric than the accuracy metric because the latter only evaluates the number of correct predictions without considering the degree of belief or probability for each prediction. For this evaluation, we bin the stars into three specific $\Delta \nu$ ranges according to the occurrence of particular red giant populations: mostly RGB ($\Delta \nu > 9 \mu$Hz), RGB and the secondary clump ($5 \mu$Hz $< \Delta \nu \leq 9 \mu$Hz), and RGB and the red clump ($\Delta \nu \leq 5 \mu$Hz). After tuning, the prediction weights of the combined model, $p_{comb}$, are defined as follows:

\[ p_{comb} = \begin{cases} 0.5p_{add} + 0.5p_{org} & \text{for } \Delta \nu \leq 5 \mu\text{Hz} \\ 0.4p_{add} + 0.6p_{org} & \text{for } 5 \mu\text{Hz} < \Delta \nu \leq 9 \mu\text{Hz} \\ 0.75p_{add} + 0.25p_{org} & \text{for } \Delta \nu > 9 \mu\text{Hz} \end{cases} \]  

where $p_{org}$ and $p_{add}$ are the predictions of the original and addition models, respectively. We show the log loss comparison between models based on 4-year data in Figure 4. The comparison of log loss values for models of shorter data lengths are very similar to that of the 4-year models and hence not shown. By comparing the log loss between the addition and original models, the addition model has a significantly smaller log loss for $\Delta \nu > 9 \mu$Hz (red), which is due to the use of the $\Delta \nu$ prior. The addition model also performs slightly better than the original model for $\Delta \nu \leq 5 \mu$Hz (blue), but performs slightly worse for $5 \mu$Hz $< \Delta \nu \leq 9 \mu$Hz (green). The relatively poor performance of both original and additional models in the range $5 \mu$Hz $< \Delta \nu \leq 9 \mu$Hz can be attributed to the difficulty in discriminating HeB secondary clump stars ($M \geq 2.0M_\odot$) from RGB stars. This is because RGB and HeB stars within this $\Delta \nu$ range may have mixed mode patterns that resemble each other (Grosjean et al. 2014, their Figure 7).

By combining predictions from the addition and original models, we reduce the frequency of having resulting severely incorrect predictions because the likelihood of two independent classifiers both predicting very incorrectly is smaller than that of a single classifier. Furthermore, if the addition and original models predict very different probabilities to one another, the resulting output will be an intermediate probability, which produces a smaller log loss than a severely incorrect prediction probability from any one model.
thermore, by assigning weights for specific $\Delta v$ ranges such as in Equation 2, we are effectively specifying which model’s prediction to prioritize within each range. As an example, for $\Delta v > 9 \mu Hz$, we observe a significant improvement of the combined model over the original model by heavily prioritizing $p_{\text{HeB}}$ with a weight of 0.75. However, the resulting log loss of the combined model is higher than the addition model’s log loss for $\Delta v > 9 \mu Hz$, suggesting that the $p_{\text{HeB}}$ needs to be weighted more. Despite this, it can be beneficial to retain some contribution from the original model because both models learn differently from one another.

We use the combined model structure for the classifier of each data length throughout the rest of the study, hence any subsequent prediction probabilities in the study refers to $p_{\text{comb}}$. Furthermore, any subsequent use of the term ‘classifier’ refers to a classifier with the combined model structure. All classifiers are constructed with the Keras library (Chollet 2015) built on top of Theano (Theano Development Team 2016). Training utilizes a Quadro K620 GPU and the NVIDIA cuDNN library (Chetlur et al. 2014).

### 3.2 Probability Calibration

To optimize the predictions of the classifier, we attempt to calibrate the output probabilities as a post-processing step. The output is the probability for the occurrence of each class $j$, (RGB or HeB), from the softmax function in the output layer of the network, given by:

$$p(y = j|x) = \frac{e^{x^T w_j}}{\sum_{k=1}^{2} e^{x^T w_k}},$$  \tag{3}$$

where $x$ are input values to the output layer and $w$ are the weights of the output layer (see H17 for details). Output values close to 0 indicate a high RGB probability, while values close to 1 indicate a high HeB probability. These probabilities are normalized such that they sum to 1 across both classes, and can be interpreted as the confidence level of the classifier in predicting a particular class for a star. Ideally, we would like the output probabilities to be consistent with the frequency by which we observe particular classes within a sample of stars. A classifier with such a quality is known to be well-calibrated. For instance, if a well-calibrated classifier predicts $p = 0.6$ for each star within a sample dataset, we would expect that the true quantity of HeB stars within that sample will be around 60%. Neural networks in general have well-calibrated probabilities (Niculescu-Mizil & Caruana 2005), hence we aim to confirm this for our own classifiers by observing the distribution of predictions from cross-validation and measuring the corresponding Brier score (Brier 1950), defined by:

$$E_{\text{Brier}} = \frac{1}{m} \left( \sum_{i=1}^{m} (y_i - \hat{y}_i)^2 \right),$$  \tag{4}$$

where $m$ is the number of stars in the dataset, $y$ the ground truth label, and $\hat{y}$ the predicted probability. Well-calibrated probabilities have $E_{\text{Brier}}$ close to zero. In addition to the Brier score, we also plot a reliability diagram (DeGroot & Fienberg 1983) by binning the classifier predictions and plotting the mean probability, $\bar{p}$, versus the true fraction of stars which are HeB within each bin. The better the calibration of the classifier, the closer the probabilities are to an increasing diagonal line from $(0, 0)$ to $(1, 1)$ on the plot.

If the probabilities of the classifier are not well-calibrated, they may be improved by transforming the predictions using either of two popular transformations. Platt scaling (Platt 1999) transforms the probabilities using a sigmoid function, with function parameters learned using maximum likelihood methods. The second type of transformation, isotonic regression (Robertson et al. 1988), works better than Platt scaling in practice with sufficient amounts of training data. Thus, this is the method we use to calibrate our probabilities. The objective of isotonic regression is to find a monotonically increasing (isotonic) function $g$, such that (Niculescu-Mizil & Caruana 2005):

$$g = \text{argmin}_g \sum_{i=0}^{m} (y_i - z(i))^2,$$  \tag{5}$$

with the argmin taken over all isotonic functions. In practice, the Pair-Adjacent-Violators Algorithm (PAVA) (Ayer et al. 1955) solves the isotonic regression problem by finding a stepwise constant solution $g$, which maps the predicted probabilities into better calibrated ones. We perform probability calibration on predictions from 10-fold cross validation, where we calibrate on half of the predictions (calibration data) to learn $g$. We transform the probabilities of the other half of the predictions using this learned $g$ and measure their resulting Brier score. We show the reliability diagrams of each classifier in Figure 5.

We see that the classifiers generally benefit from calibration

![Figure 5. Reliability diagrams for (a) the 4-year classifier, (b) the 356-day classifier, (c) the 82-day classifier and (d) the 27-day classifier. The uncalibrated probabilities are in blue, while the probabilities calibrated with isotonic regression are in green. The Brier score of the probabilities are listed for each classifier. The dotted diagonal line represents a perfect calibration of probabilities.](image-url)
by isotonic regression, shown by calibrated probabilities (green) that follow the diagonal line of perfect calibration reasonably well and have slightly lower Brier scores compared to the uncalibrated probabilities (blue). However, there is a difficulty of applying isotonic regression to 4-year probabilities because of the sparsity of intermediate probabilities. As can be seen in Figure 6a, the 4-year classifier is highly confident in its predictions, such that most stars are placed in probability bins near 0 or 1. Due to this, the stepwise constant solution \( g \) obtained by isotonic regression becomes very coarse, which results in only a few bins for the calibrated probabilities (see Figure 5a). In contrast, the other classifiers do not experience the same level of sparsity for intermediate probabilities (e.g. Figure 6b). Thus, we adopt isotonic regression for all classifiers except the 4-year classifier.

Through the reliability diagrams in Figure 5, we can observe how each classifier’s predictions are biased relative to the true fraction of HeB stars. For instance, the output probabilities by the 4-year classifier generally overestimate the true fraction of HeB stars because the probability curve is consistently below the diagonal line except near a mean predicted value, \( \bar{p} \), of 0.6. The 82-day classifier underestimates the true fraction of HeB stars from 0.3 \( \bar{p} \) 0.7, while the calibrated 27-day classifier instead slightly overestimates the true fraction of HeB stars for the same \( \bar{p} \) range. Meanwhile, the calibrated 356-day classifier probabilities follow the diagonal line well and do not show significant bias across \( \bar{p} \).

### 3.3 Choosing a Probability Threshold

Instead of using \( p_{\text{thres}} = 0.5 \) as a fixed probability threshold for determining if a star is RGB \( (p < p_{\text{thres}}) \) or HeB \( (p \geq p_{\text{thres}}) \), we determine an ideal threshold that will maximize the accuracy of the classifier. We perform this by plotting the Receiver Operating Characteristic (ROC) curve (Swets et al. 2000). As shown in Figure 7, the ROC curve plots the true positive rate (TPR) against the false positive rate (FPR) of the classifier, defined as follows:

TPR is the ratio of correct HeB predictions to all stars that are truly HeB. Here it is the classifier’s ability to find all HeB stars. This is also known as recall.

FPR is the ratio of stars incorrectly predicted as HeB to all stars that are truly RGB.

The ROC curve begins at the origin in TPR-FPR space. Prediction probabilities for the evolutionary state of the stars are ranked in descending order, and this determines the order in which they are plotted on the ROC curve (Flach & Matsubara 2008). Beginning at scores close to 1, if this prediction correctly corresponds to a HeB star, we plot vertically upwards with magnitude 1/TPR. Otherwise, if the true label is actually RGB, we plot horizontally to the right with magnitude 1/FPR. Thus, a perfect classifier’s ROC curve will plot all the true HeBs first (a vertical line upwards all the way to 1 on the ordinate), followed by all the true RGBs (horizontal line to the right). The ROC curve illustrates how well the classifier separates HeB stars from RGB stars, with the classifier performing better the closer the ROC curve is to the upper left corner (Zweig & Campbell 1993), corresponding to an Area Under Curve (AUC) closer to a value of 1 as described in Section 2.3. Following the method described by Flach (2012), we can use the ROC curve to determine the threshold that best separates the classes for each classifier. We first determine the accuracy isometrics of the classifier,

---

**Figure 6.** A typical distribution of predicted scores for the (a) 4-year classifier and the (b) 27-day classifier from 10-fold cross validation.

**Figure 7.** ROC curve of the classifier on 10-fold cross validation of 4-year Kepler data. The black, dashed diagonal line corresponds to random guessing for predictions, while the red line is the accuracy isometric. A few points of the curve are plotted (blue dots) with their corresponding \( p \) to illustrate the order of plotting. (Inset) A close-up of the plot near the point of intersection (red dot) of the accuracy isometric with the ROC curve. This point corresponds to a prediction score \( p_{\text{thres}} = 0.712 \), which is the ideal score threshold.
which are lines in TPR-FPR space with gradients equal to the ratio of RGB stars to HeB stars. We find the highest accuracy isometric that can intersect the ROC curve, which results in a single point of intersection between the accuracy isometric and the ROC curve (see Figure 7). Because each point on the ROC curve is plotted in descending order of probability, the point of intersection between the ROC curve and the accuracy isometric corresponds to a probability, \( p_{\text{thres}} \), which is the best classifier probability threshold for separating RGB and HeB populations from one another.

In Table 1, we summarize the determination of the ideal probability thresholds (see Figure 7) for separating RGB and HeB stars, and the presence of post-processing calibration (Section 3.2) for all of our trained classifiers. In our study, we assign equal importance to predictions of both RGB and HeB stars. As such, the ideal choice of a probability threshold should be \( p_{\text{thres}} = 0.5 \). We generally obtain values close to 0.5 except for the 4-year data, which has \( p_{\text{thres}} = 0.712 \). We can interpret a probability threshold of 0.712 as follows: we only assign the HeB label to a star if it is at least 71.2% certain that it recognizes the input spectra to be that of a HeB star.

Interestingly, there is a correlation between the determination of an ideal probability threshold from the ROC curve and how well the probabilities are calibrated. Using Figure 8 as an example, we see that by measuring the abscissa of the largest intersection of the true HeB fraction = 0.5 with the reliability diagram, we find that the corresponding \( \hat{p} \) is approximately equal to \( p_{\text{thres}} \). As a consequence, better calibrated probabilities have \( p_{\text{thres}} \) closer to 0.5. This is shown by reliability curves that follow the diagonal better (e.g. Figure 8b) having \( p_{\text{thres}} \) closer to 0.5 as compared to those that do not (e.g Figure 8a). This correlation also implies that we can obtain the best classification accuracy for our classifier if we use a threshold \( p \), which splits a random sample of stars into equal parts HeB and RGB.

Table 1. Ideal probability thresholds and the use of post-processing calibration for classifiers corresponding to each length of data.

<table>
<thead>
<tr>
<th>Data Length</th>
<th>4 years</th>
<th>351-day</th>
<th>82-day</th>
<th>27-day</th>
</tr>
</thead>
<tbody>
<tr>
<td>Ideal Threshold (( p_{\text{thres}} ))</td>
<td>0.712</td>
<td>0.588</td>
<td>0.459</td>
<td>0.586</td>
</tr>
</tbody>
</table>

Table 2. Performance metrics for each classifier on their corresponding test set. Uncertainties are shown in compact bracket form: e.g., 0.981 (2) = 0.981 ± 0.002, 0.110 (17) = 0.110 ± 0.017.

<table>
<thead>
<tr>
<th>Data Length</th>
<th>4 years</th>
<th>356 days</th>
<th>82 days</th>
<th>27 days</th>
</tr>
</thead>
<tbody>
<tr>
<td>Accuracy</td>
<td>0.981 (2)</td>
<td>0.984 (3)</td>
<td>0.955 (3)</td>
<td>0.934 (3)</td>
</tr>
<tr>
<td>Precision</td>
<td>0.981 (3)</td>
<td>0.982 (4)</td>
<td>0.956 (3)</td>
<td>0.934 (3)</td>
</tr>
<tr>
<td>Recall</td>
<td>0.982 (2)</td>
<td>0.982 (3)</td>
<td>0.954 (3)</td>
<td>0.933 (3)</td>
</tr>
<tr>
<td>F1 Score</td>
<td>0.982 (2)</td>
<td>0.982 (3)</td>
<td>0.954 (3)</td>
<td>0.933 (3)</td>
</tr>
<tr>
<td>ROC AUC</td>
<td>0.998 (1)</td>
<td>0.995 (1)</td>
<td>0.988 (1)</td>
<td>0.981 (1)</td>
</tr>
<tr>
<td>Log Loss</td>
<td>0.044 (4)</td>
<td>0.060 (15)</td>
<td>0.184 (21)</td>
<td>0.174 (7)</td>
</tr>
<tr>
<td>Brier</td>
<td>0.010 (1)</td>
<td>0.012 (1)</td>
<td>0.034 (2)</td>
<td>0.049 (1)</td>
</tr>
</tbody>
</table>

4 RESULTS OF THE NEW CLASSIFIER

Using the methods described in the previous section, we first predict on test sets for each data length. Then we make new predictions on the ‘full mission’ Kepler red giants, including some stars without any prior seismic classification. Next, we apply the classifier on the K2 data of the open cluster M67. As opposed to H17, we now also determine uncertainties on the classification predictions, which we describe in Section 4.1.

4.1 Uncertainties for Classifier Predictions

When predicting on test sets or on real datasets, we cannot use the standard deviation across cross-validation folds as our uncertainty. Instead, we measure the uncertainty of each prediction produced by the classifiers using Monte Carlo dropout to approximate a Gaussian process (Gal & Ghahramani 2016). Dropout is an effective way to prevent overfitting by randomly setting neurons within the neural
network to zero with a certain probability (Srivastava et al. 2014). It is often implemented differently when the classifier is being trained as opposed to when it is predicting on test data. During training, dropout is implemented by randomly setting neurons to zero with a certain drop out probability, but during testing, it is implemented by scaling the weights for these neurons with the value of the dropout probability. By passing data through the network multiple times during testing with the ‘training’ form of dropout enabled, we effectively perform a Monte Carlo integration over a Gaussian process posterior approximation (Gal & Ghahramani 2016). Hence, we report classifier predictions as the expectation of 50 passes through the network, with the standard deviation as the prediction uncertainty.

4.2 Test Set Performance

The performance of the classifiers on their respective test sets are shown in Table 2. In general, the shorter the data length, the lower the classifier performance. This is expected because the folded spectra of shorter data lengths have lower frequency resolution. A notable exception to this is the performance for the 356-day classifier. This shows that 356-day data still have the level of detail as the 4-year data for the purpose of classification (see Figure 1). Furthermore, because the 356-day classifier trains on nearly three times more stars than the 4-year classifier, the network displays a greater robustness and hence a better accuracy than the 4-year classifier.

4.3 Revised Kepler Unclassified Set Predictions

We provide new or updated predictions for 8633 Kepler red giants with $\Delta \nu > 2 \mu$Hz and no prior asymptotic period spacing measurements, from which 7655 were already classified in H17 as part of the unclassified set. This new set of 8633 red giants contains 426 red giants that have not been given any seismic-based evolutionary state classifications prior to this study (not even by H17). We present our results in a comprehensive table (available online), which includes evolutionary state classifications for the 6350 stars in our training set. In addition to the 14983 stars that are classified, a sample of this catalogue is shown in Table 3.

From a comparison of old (H17) and new (this work) classifications on the unclassified set in Figure 9, it can be seen that our new model now correctly does not predict any HeB stars at $\Delta \nu > 10 \mu$Hz. In addition, there are several previously classified HeB stars at $\Delta \nu \approx 5 \mu$Hz with $\epsilon \gtrsim 1.3$ and at $\Delta \nu < 7 \mu$Hz with $\epsilon \lesssim 0.8$ (Figure 9a) that are now reclassified as RGB stars with the new probability threshold ($p_{\text{thres}} = 0.712$) (Figure 9b). However, these newly classified RGB stars are assigned $p \approx 0.6$, such that the classifier only weakly believes they are on the RGB. Nonetheless, it may be worthwhile in future work to investigate these predicted RGB stars with low $\Delta \nu$ that appear to deviate from the $\Delta \nu - \epsilon$ relation of RGB stars (Kallinger et al. 2012).

Generally, more symbols in Figure 9b have lighter colours than in Figure 9a, which indicate that more stars have probabil-
Table 4. Predictions on M67 giants from K2 data with $\Delta \nu > 2.8\mu$Hz using the 82-day classifier ($p_{\text{thres}} = 0.459$), with classifier probabilities $p$. Values for $\Delta \nu$, $\nu_{\text{max}}$, and colour-magnitude diagram (CMD) classifications are obtained from Stello et al. (2016b). Disputes are highlighted in bold.

| EPIC ID | $p(1|\nu)$ | Classifier | CMD |
|---------|-------------|------------|-----|
| 211406540 | 0.999 (0.001) | HeB | HeB | 31.8 | 4.59 |
| 211406541 | 0.858 (0.131) | HeB | RGB | 34.7 | 4.27 |
| 211418433 | 0.998 (0.002) | HeB | HeB | 36.5 | 4.18 |
| 211405253 | 0.841 (0.165) | HeB | HeB | 37.8 | 3.95 |
| 211415732 | 0.999 (0.001) | HeB | HeB | 37.9 | 4.40 |
| 211420284 | 0.983 (0.017) | HeB | HeB | 39.3 | 4.22 |
| 211413402 | 0.619 (0.211) | HeB | HeB | 39.5 | 4.44 |
| 211417056 | 0.972 (0.022) | HeB | HeB | 39.6 | 4.17 |
| 211392837 | 0.048 (0.072) | RGB | RGB | 47.5 | 4.81 |
| 211413623 | 0.005 (0.011) | RGB | RGB | 64.8 | 6.28 |
| 211396385 | 0.005 (0.007) | RGB | RGB | 77.4 | 7.00 |
| 211414300 | 0.533 (0.214) | HeB | RGB | 78.8 | 7.19 |
| 211408546 | 0.116 (0.098) | RGB | RGB | 98.7 | 8.17 |
| 211410231 | 0.061 (0.070) | RGB | RGB | 103.1 | 8.87 |
| 211412928 | 0.001 (0.001) | RGB | RGB | 117.8 | 9.74 |
| 21141629 | 0.001 (0.001) | RGB | RGB | 196.0 | 14.43 |
| 211414687 | 0.001 (0.001) | RGB | RGB | 203.0 | 15.10 |
| 211416749 | 0.001 (0.001) | RGB | RGB | 243.3 | 16.76 |
| 211421954 | 0.001 (0.001) | RGB | RGB | 246.1 | 17.47 |
| 211409560 | 0.001 (0.001) | RGB | RGB | 272.2 | 19.10 |
| 211388537 | 0.004 (0.006) | RGB | RGB | 287.6 | 20.15 |
| 211403248 | 0.000 (0.002) | RGB | RGB | 305.5 | 21.45 |
| 211415364 | 0.005 (0.020) | RGB | RGB | 463.0 | 28.29 |
| 211419224 | 0.054 (0.069) | RGB | RGB | 559.0 | 36.34 |
| 211409088 | 0.028 (0.044) | RGB | RGB | 562.0 | 33.02 |

* We use short cadence data for this star.

4.4 M67 Red Giant Predictions

To test our 82-day classifier on real K2 data, we obtain the background-corrected K2 spectra along with measurements of $\Delta \nu$ and $\nu_{\text{max}}$ of red giants within the open cluster M67 from the study by Stello et al. (2016b). From the cluster, we predict the evolutionary state of 25 red giants with $\Delta \nu > 2.8\mu$Hz, following the $\Delta \nu$ range limitation of our classifier. We compare our asteroseismically-derived classifications with the evolutionary state classifications from the colour-magnitude diagram (CMD). The outcome is tabulated in Table 4. Our classifications are in good agreement with those inferred from the colour-magnitude diagram (CMD). Only two red giants have disputed classifications, where the classifier believes both are HeB. However, in general, the agreements indicate a good classifier performance.

Reporting the uncertainties of each prediction shows which stars are the most difficult to classify. Large uncertainties imply the classifier is highly uncertain about its degree of belief. If the resulting $p$ is close to the threshold $p_{\text{thres}} = 0.459$, this then implies that the classifier is highly uncertain on the resulting predicted class. In Table 4, we identify the disputed star EPIC 211414300 showing such an uncertain prediction. Interestingly, we see that the classifier is able to correctly predict the evolutionary states for RGB stars with $\nu_{\text{max}} > 280\mu$Hz despite only having trained on long cadence data. This is because the additional $\Delta \nu$ input (Figure 3) allows the classifier to be able to generalize its predictions to red giants above the long cadence Nyquist frequency ($\nu_{\text{max}} \lesssim 280\mu$Hz).

5 Robustness of the New Classifier

To test the robustness of our deep learning classifiers, we determine the impact of suboptimal training or testing conditions on classifier performance.

5.1 Flipping the Truth Labels of Training Data

We aim to simulate the scenario in which assigned training labels may be unreliable by randomly flipping population labels of stars in the training set from HeB to RGB and vice versa during the process of training the classifier. The classifier performance is measured using 10-fold cross validation because training with suboptimal data may cause large variations in performance. We randomly flip a percentage of training labels during training, from 0% to 100% in 10% increments. Then we make predictions on the validation fold, which does not have any stars with flipped labels. This analysis illustrates the dependence of the classifier performance on the accuracy of its ground truth during training.

Figure 10a shows the behaviour of the classifier trained on 4-year data as we introduce incorrectly labelled, or label-flipped, ground truth into the training set. The classifier is remarkably robust, with the accuracy remaining almost constant until about 40% incorrect training labels. Thus, the predictions are generally resistant to wrong information learned during training as long as the majority of training examples are correct. Meanwhile, the log loss shows the expected monotonically increasing behaviour indicative of an increase in uncertainty as more examples in the training set are incorrect. The very high log loss at a 100% of incorrect labels.
in the training set indicates that the classifier is now confidently giving very incorrect predictions.

By examining the decomposition of the classifier’s accuracy according to $\Delta \nu$ regions as defined in Equation 2, we can reveal interesting prediction behaviours of the classifier. We show this for the 4-year classifier in Figure 10b. We see that all curves are approximately symmetrical about the 50% flipped label ordinate, with different $\Delta \nu$ regions having different accuracies and uncertainty ranges as the proportion of incorrect training data varies. Because the accuracy and log loss behaviours (as of Figure 10a) for the classifiers of shorter data lengths are similar to that of the 4-year classifier, we do not show them. However, we do present their accuracy curve decompositions in Figure 11. Due to the symmetry of the curves, we only plot up to 50% of flipped training labels.

Similar to the 4-year classifier in Figure 10b, the 356-day, 82-day, and 27-day classifiers require at least $\approx 40\%$ of incorrect training data before they experience a significant impact in performance (drop in accuracy greater than 10%). Next, we see that although the curve for $\Delta \nu > 9 \mu$Hz (red) for all classifiers has the highest accuracy for low proportions of incorrect training data, it develops a high level of uncertainty (broad red band) from about a 30% proportion of incorrect training data onwards. In this $\Delta \nu$ range, the training set mainly contains RGB stars. Providing a fraction of incorrect training data in this $\Delta \nu$ range forces the classifier to learn a rule that allows it to split very similar images with similar $\Delta \nu$ into two separate classes. The difficulty of this induces a large degree of uncertainty in predictions. In contrast, for other regions where both true HeB and RGB stars exist, it may be easier for the classifier to learn a rule to split the data into two classes as long as there exists a significant proportion of training data that are assigned different classes and are not too identical to one another. In support of this, we see from Figures 10 and 11 that the accuracy curves for $\Delta \nu \leq 5 \mu$Hz (blue) generally only develop high uncertainties near a 50% proportion of flipped training labels. We also note that across all classifiers, the accuracy curves for $5 \mu$Hz $< \Delta \nu < 9 \mu$Hz (green) are consistently the lowest in accuracy for low proportions of flipped training labels, which agrees with our result in Figure 4. Although this is the case, we see that the green curves do not develop very large uncertainties as the proportion of incorrect training data increases. It is likely that this is related to the intrinsic difficulty of classifying RGB and HeB stars at that range of $\Delta \nu$.

5.2 Added White Noise in Test Data

In previous sections, we have assumed that the test set is of similar quality to the training set. In reality, this is not always the case, because future datasets may be very different from Kepler data in terms of quality, though we would still want to use Kepler data to train classifiers for K2 and TESS data for instance. Hence, we study the tolerance of the classifier performance when predicting on spectra in the test set with ‘distortions’ in the form of added white noise. This analysis will measure how well the classifier performs at lower signal-to-noise ratios.

We use power spectra divided by the background noise, such that the mean noise level is 1.0. To simulate additional white noise, we add specific levels of noise sampled from a $\chi^2$ distribution onto the entire power spectrum. We first define the signal level of the power excess as the median power of the 30 highest power frequency bins within a $4 \Delta \nu$ range around $\nu_{\max}$. We control the mean noise level that will be added to the spectrum by dividing this median signal by a factor $R$. As an example, using $R = 1$, the mean noise level that will be added is the same as the initial signal level. We then add this mean noise level to the spectra and quantify the resulting spectra quality using a signal-to-noise ratio, $S_N$, which is the new signal level of the spectrum, divided by the new mean noise level. Thus, by definition, $S_N$ is similar to $R$, except that it is measured after adding white noise to the spectra (see Figure 12). We use $R$ with values of 1, 5, 10, 15, up to 50 in steps of 5 to generate noisier versions of the test sets. We then test the classifier performance on these noisier test sets and plot the performance versus binned $S_N$ values in Figure 13.

We see that the classifiers are robust towards noise by generally maintaining accuracies above 90% for $S_N \geq 10$. Below $S_N \approx 10$, as seen in Figure 12, is where the white noise level starts obscuring oscillation modes of lower amplitude in both the power spectrum and the folded spectrum. Classification at such low $S_N$ is thus difficult for both the expert eye and the machine expert. Nonetheless, we partially attribute this overall robustness towards white noise levels to the use of folded spectra. Because oscillation modes in the folded spectra generally overlap one another, they can still be distinctly observed even in the presence of large amounts of white noise (see lower rows in Figure 12). Hence, the classifiers can still detect them even on data with lower frequency resolution and can still make reasonably accurate predictions at low $S_N$ values as seen in Figures 13c and 13d. Interestingly, we see that the 4-year and 356-day classifiers do not achieve the test set accuracies in Table 2 in the limit of high $S_N$. One possible explanation is that a significant number of stars that are difficult to classify have been placed into such $S_N$ bins. An alternate possibility is the presence of adversarial examples (Szegedy et al. 2013), where small pertur-

---

**Figure 11.** The decomposition of accuracy curves for (a) 356-day, (b) 82-day, and (c) 27-day data as a function of the number of incorrect training data by 10-fold cross validation. The shaded regions are $1\sigma$ uncertainty ranges. To be concise, proportions of flipped labels only up to 50% are plotted due to the symmetry of the accuracy curves about this ordinate.
Improved Asteroseismic ConvNets for K2/TESS

Figure 12. Noise normalised power spectra (left) and folded spectra (right) of 4-year Kepler data of RGB star KIC 8144802, after adding white noise. The resulting quality of the spectra is quantified by a measured signal-to-noise ratio, $S_N$, which determines how many times the signal level (black line) is greater than the noise level (red line). The amount of added noise to the original spectra is defined as the signal level prior to adding noise, divided by the factor $R$.

Figure 13. Accuracy of each trained classifier on test sets with added white noise, as a function of the resulting observed signal-to-noise ratio, $S_N$. The histograms in each plot show the fraction of test stars within each $S_N$ bin.

Figure 5.3 $\Delta \nu$ Variations

While we have accounted for less precise $\Delta \nu$ values in Section 2.2, it is still insightful to investigate how the classifiers perform on stars with $\Delta \nu$ values that deviate by a fixed fraction of their 'correct' 4-year value. Doing this provides us with a measure of classifier robustness against varying $\Delta \nu$ uncertainties. Furthermore, this allows us to investigate the forward compatibility of the classifier with respect to $\Delta \nu$, such that we can form a reasonable estimate of how much better or worse the classifiers can perform when predicting on a dataset with better or worse $\Delta \nu$ uncertainties than that shown in Figure 2.

We first perturb the 4-year $\Delta \nu$ of each star, $\Delta \nu_{4yr}$, in the test set by a fixed fraction of its value, and denote the resulting value as $\Delta \nu_p$. While the perturbation magnitude is fixed, its direction is random, such that the perturbation is either added or subtracted. We perturb copies of test sets with $\Delta \nu_p$ fractional differences of 0.1%, 0.2%, 0.5%, 1%, 2%, 5%, and 10%. We note that while the 356-day, 82-day, and 27-day test sets that we have previously constructed in this study have used less precise $\Delta \nu_p$ values with uncertainty distributions shown in Figure 2, the corresponding test sets that we construct in this Section initially use $\Delta \nu_{4yr}$ that we then perturb. We plot the classifier performance on these perturbed test sets in Figure 14. As expected, more perturbed $\Delta \nu$ values result in worse classifier performances. We show the effect of perturbed $\Delta \nu$ values on 4-year folded spectra in Figure 15. From the Figure, we see that for...
\(\Delta \nu \gtrsim 1.01 \Delta \nu_{4 \text{yr}}\), oscillation modes (features) of the same spherical degree, \(l\), do not overlap each other well, causing a more complicated mode distribution in the folded spectrum, which potentially confuses the classifier. This agrees with the significant drop in performance for the 4-year classifier in Figure 14a at \(\Delta \nu \gtrsim 1.01 \Delta \nu_{4 \text{yr}}\). However, we see in Figures 14b-d that classifiers of shorter data lengths are much more robust towards \(\Delta \nu\) perturbations than the 4-year classifier, because this is notable that small accuracies do not decrease as easily with \(\Delta \nu\). There are two reasons for this, with the first being the lower frequency resolution of shorter data lengths, which makes the effect of small \(\Delta \nu\) perturbations less pronounced visually on their folded spectra than on the 4-year-based folded spectra. Secondly, the classifiers of shorter data lengths are more likely to have ‘seen’ folded spectra made with less precise \(\Delta \nu\) values because they trained on data using \(\Delta \nu\) with higher uncertainties than the 4-year \(\Delta \nu\) measurements. In contrast, the 4-year classifier has trained using \(\Delta \nu\) values with small uncertainties and thus experiences difficulties generalizing to \(\Delta \nu\) values with a large uncertainty.

When the adopted \(\Delta \nu\) is close to \(\Delta \nu_{4 \text{yr}}\), the 356-day, 82-day, and 27-day classifiers notably achieve higher test accuracies (99.0%, 97.1%, and 94.8%, respectively) than their test accuracies listed in Table 2, which were based on more uncertain, but more realistic \(\Delta \nu\) values. This suggests that these classifiers are using the right features for classification because they can better recognize features that overlap well in the folded spectra. Interestingly, we note that the accuracy for the 4-year classifier does not monotonically decrease (or increase for log loss) with increasing \(\Delta \nu\). At \(\Delta \nu \approx 1.1 \Delta \nu_{4 \text{yr}}\), the classifier accuracy increases (or decreases for log loss), which may be due to the overlapping of certain modes, such as \(l = 0\) with \(l = 2\), at particular fractional \(\Delta \nu\) differences being correctly recognized by the classifiers.

6 CONCLUSIONS

We have developed a variant of our previous deep learning classifier, which we call the combined classifier. The combined classifier improves the prediction accuracy of our previous classifier and avoids predicting HeB stars at high \(\Delta \nu\). We trained combined classifiers for 4-year, 356-day, 82-day, and 27-day photometric timeseries, which are representative for Kepler, K2, TESS, and large parts of the PLATO sample. In order to optimize the combined classifier’s performance, we calibrated the probabilities of each classifier using isotonic regression. We adopted this post-processing for all classifiers except the 4-year classifier. In addition, we determined the probability threshold \(p_{\text{thresh}}\) that maximizes the classifier accuracy. As a result, we could report test set accuracies of 98.1% (4 years), 98.4% (356 days), 95.5% (82 days), and 93.4% (27 days).

Next, we presented the evolutionary state classifications of 14983 Kepler red giants using the 4-year classifier. While 5000 of these were training stars, our classifier made predictions on 1350 test stars and the remaining 8633 stars. The classifications for these 8633 stars include 426 that previously had no seismic classification at all. We also predicted the evolutionary states of red giants within the open cluster M67 using the 82-day classifier on K2 data. The predictions showed good agreement with classifications derived using the colour-magnitude diagram, with only two disputed predictions.

We then analysed the robustness of the classifiers, and found that they are remarkably robust towards incorrect ground truths, needing approximately 40% of training data to be incorrect before
the classifier performance is significantly affected. We also found that the classifiers can still make accurate predictions in the presence of significant white noise levels. By testing the classifiers on folded spectra constructed using perturbed Δν values, we found that classifiers that trained with uncertain Δν values were highly robust towards perturbed Δν values. In addition, high test accuracies in the limit of small Δν perturbations suggest that the classifiers detect the right set of discriminating features for classification. The accuracy of the classifiers along with the ability to learn discriminating features even for low resolution or noisy data proves that deep learning, as an approach to artificial intelligence, provides a very powerful and robust method to perform asteroseismic classification using power spectra.

ACKNOWLEDGEMENTS

Funding for this Discovery mission is provided by NASA’s Science Mission Directorate. We thank the entire Kepler team without whom this investigation would not be possible. D.S. is the recipient of an Australian Research Council Future Fellowship (project number FT1400147). We would also like to thank Timothy Bedding, Daniel Huber, and the asteroseismology group at The University of Sydney for fruitful discussions.

REFERENCES

Borucki W. J., et al., 2010, Science, 327, 977
Brier G. W., 1950, Monthly Weather Review, 78, 1
Chollet F., 2015, Keras, https://github.com/fchollet/keras
Rauer H., et al., 2014, Experimental Astronomy, 38, 249
Robertson T., Wright F. T., Dykstra R., 1988, Order Restricted Statistical Inference. John Wiley & Sons
Simonian Y., Zisserman A., 2014, CoRR, abs/1409.1556
Zweig M. H., Campbell G., 1993, Clinical Chemistry, 39, 561

This paper has been typeset from a TEX/LaTEX file prepared by the author.