MNRAS **000**, 1–5 (2017) Preprint 22 March 2017 Compiled using MNRAS LATEX style file v3.0

Deep Learning Classification in Asteroseismology

Marc Hon,[1]⋆ Dennis Stello,[1,2] and Jie Yu[2]
[1]*School of Physics, The University of New South Wales, Sydney NSW 2052, Australia*
[2]*Sydney Institute for Astronomy (SIfA), School of Physics, University of Sydney, NSW 2006, Australia*

Accepted XXX. Received YYY; in original form ZZZ

ABSTRACT
In the power spectra of oscillating red giants, there are visually distinct features defining stars ascending the red giant branch from those that have commenced helium core burning. We train a one-dimensional convolutional neural network by supervised learning to automatically learn these visual features from images of folded oscillation spectra. By training and testing on *Kepler* red giants, we achieve an accuracy of up to 99% in separating helium-burning red giants from those ascending the red giant branch. The convolutional neural network additionally shows capability in accurately predicting the evolutionary states of 5379 previously unclassified *Kepler* red giants, by which we now have greatly increased the number of classified stars.

Key words: asteroseismology – methods: data analysis – techniques: image processing – stars: oscillations – stars: statistics

1 INTRODUCTION

A key concept in determining stellar ages of red giants is distinguishing the evolutionary state i.e. classifying between stars ascending the red giant branch (RGB) and those that have commenced core helium burning (HeB). Space missions such as *Kepler* (Borucki et al. 2010) and the upcoming TESS (Ricker et al. 2014), have and are providing an enormous quantity of red giant oscillation spectra, which makes manual or semi-automatic classification of the population class of each star infeasible. Automated methods do exist (e.g Vrard et al. (2016)), however considerable effort is required into defining and acquiring features such as the period spacing ΔP (Bedding et al. 2011; Mosser et al. 2014) or the structure of mixed modes (Elsworth et al. 2016) in order to separate the populations. Furthermore, these methods require relatively high signal-to-noise data.

Here, we present a deep learning method that allows spectral features to be learnt by the machine using convolutional neural networks. These are machine learning methods that mimic biological neuron structures, aimed towards feature detection in data (Fukushima 1980). They have achieved significant success over the past few years in computer vision methods such as image recognition (Sermanet et al. 2012) and even facial recognition (Garcia & Delakis 2004).

We introduce the concept of representing the oscillation frequency spectra of stars as *images* as opposed to a series of values. These images are simple representations of the power excess, which in principle contain sufficient visual features for RGB-HeB classification. By learning from an existing set of classified red giants based on ΔP measurements, we use 1-D convolutional neural networks as a form of supervised machine learning aimed to automatically learn features separating RGB from HeB stars in order to make fast yet accurate classification predictions on vast amounts of yet unclassified stars.

⋆ E-mail: mtyh555@uowmail.edu.au

2 METHODS

Here we describe the preparation of the image representation known as a *folded spectrum*, along with an overview of convolutional neural networks, and the construction of a deep learning classifier to classify the image representation.

2.1 Data

We obtain the evolutionary state classifications of 5673 *Kepler* stars based on automated asymptotic period spacing measurements by Vrard et al. (2016, hereafter Vrard), and add 335 stars from the classification by Mosser et al. (2014, hereafter Mosser) that are not already in Vrard's sample. We then assign RGB stars with the binary class 0 and HeB stars with class 1. About 30% were RGB stars. We randomly choose 1008 stars as test data, with the remaining 5000 stars for training. Additionally, we have an *unclassified set* comprising 8794 *Kepler* red giants that are known to oscillate but have not been given classifications by Vrard or Mosser. However, 517 stars in our unclassified set have been classified by Stello et al. (2013) and 2475 by Elsworth et al. (2016), of which 232 stars are in common to both. The remaining stars are yet unclassified due to the limitations of previous classification methods. We want to predict the population labels of all stars in our unclassified set.

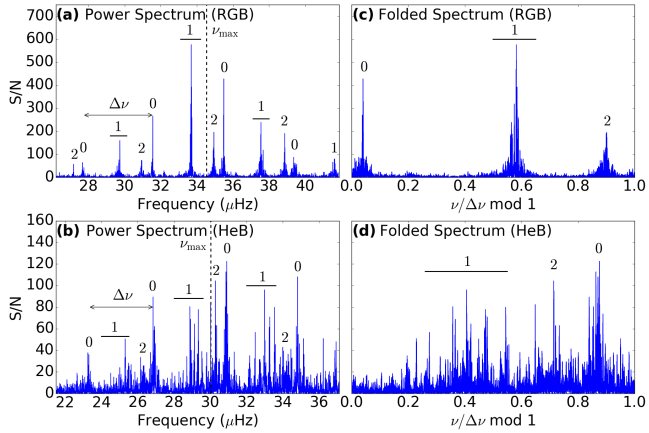


Figure 1. Comparisons between an RGB star KIC 11293804 (top), with a HeB star KIC 5810333 (bottom), both having large frequency spacing $\Delta\nu \simeq 3.92\mu$Hz. (a) and (b) are the original power spectra, while (c) and (d) are *folded spectrum* image representations. The oscillation modes are labelled by their degree l, while the frequency of maximum oscillation power, $\nu_{\rm max}$, is indicated by the dashed vertical line.

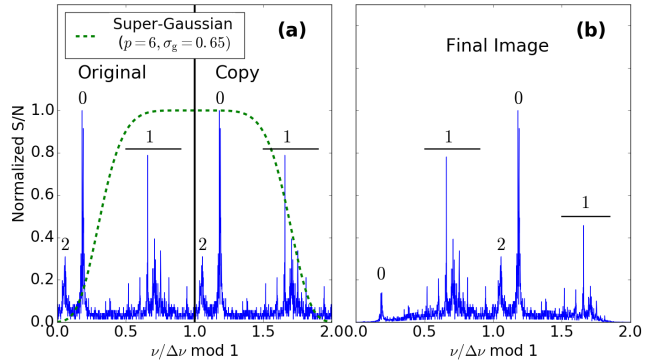


Figure 2. (a) Appended and normalized folded spectrum of KIC 10790301 with mode identification. p is the shape parameter (=2 for standard Gaussian), while $\sigma_{\rm g}$ is the standard deviation of the super-Gaussian weight. The solid vertical line separates the original image from its appended copy. (b) Resulting image from application of the super-Gaussian weight to the spectrum in (a).

2.2 Image Representation

As our image representation, we define the *folded spectrum* as the $4\Delta\nu$-wide power spectrum segment centred at $\nu_{\rm max}$, folded by a length of $\Delta\nu$(see Figures 1a, c). The spectra and values for $\Delta\nu$ and $\nu_{\rm max}$ were derived from end-of-mission *Kepler* data using the SYD pipeline (Huber et al. 2009, Yu et al., in prep.). Because the neural network requires a fixed input array length, we bin each folded spectrum into 1000 bins.

A comparison of spectral image representations between RGB and HeB stars are shown in Figures 1c, d. RGB stars clearly exhibit acoustic modes that are highly localized (Figures 1a, c) while HeB stars show broader mode distributions particularly for non-radial modes because of the stronger coupling between core and envelope (Figures 1b,d) (Dupret et al. 2009). With acoustic resonances less localized, HeB spectral representations notably have greater visual complexity as compared to RGB spectra. Besides the structure of modes, the location of the $l = 0$ mode, represented by ϵ, can be a strong indicator in distinguishing population classes (Kallinger et al. 2012). However, ϵ is not the sole feature that is used to recognize population classes from an image. The lack of a clear boundary separating the two evolutionary states shown by the observed spread in ϵ (Kallinger et al. 2012) and from theoretical studies (Christensen-Dalsgaard et al. 2014) makes ϵ unsuitable as a sole selection criterion. However, information about ϵ complements features extracted from mixed modes in the image.

As image pre-processing, we normalize each spectrum by its max power value. Then, to avoid edge effects, we append the image with a copy of itself and apply a super-Gaussian window function as shown in Figure 2.

2.3 Convolutional Neural Networks

The elementary structure of the neural network is a mathematical representation of a neuron with multiple input neurons (Figure 3a). The total input to a neuron, or node, is given by $\mathbf{w} \cdot \mathbf{x}$, with $\mathbf{x}$= $(x_1, x_2, x_3, ..., x_n)$ as an input vector with n number of features from the input layer (represented by the power in each frequency bin in our study). $\mathbf{w}$= $(w_0, w_1, w_2, ..., w_n)$ is the weight vector linking each input to a node in the subsequent layer, with w_0 known as the input bias, b, which is analagous to the intercept in a linear regression.

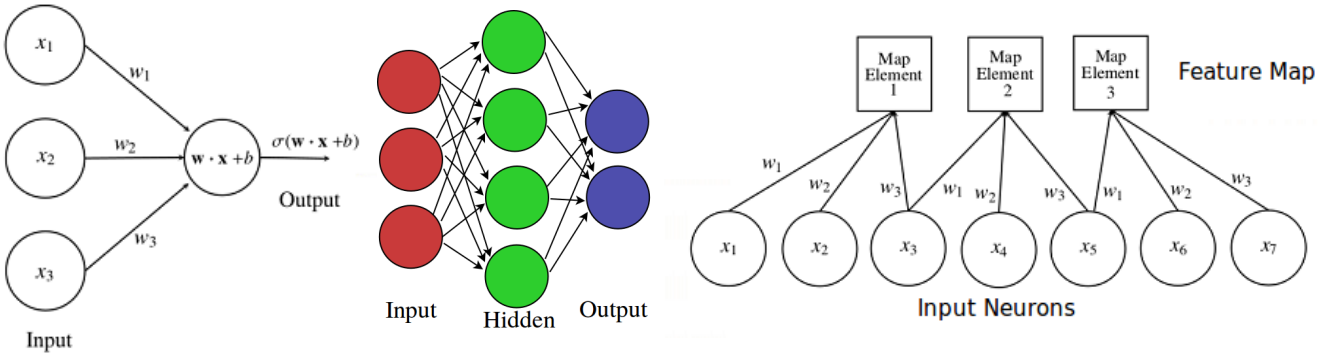
The total input is linear, however through an *activation function* (Rosenblatt 1962), σ, non-linear representations are computed between layers of the network. In this study, we use the *rectified linear unit* activation function $\sigma(x) = \max(0, x)$ suited for feature learning in neural networks (Nair & Hinton 2010), for every neural network layer except the output layer. We use feedforward neural networks where the inputs are computed and fed forward through consecutive intermediate or *hidden* layers to reach the output layer. A typical feedforward neural network has fully-connected layers, which maximizes the number of distinct nodal connections with the following layer (Figure 3b).

Convolutional neural networks (LeCun & Bengio 1998) are a variant of feedforward neural networks in which the layer connections are constrained to be local. This is achieved by *weight sharing*, where weights across neurons are constrained to a fixed set of values and are run over the neurons sequentially (Figure 3c). A fixed set of weights is known as a filter, which picks out a specific feature from the data to be stored in a *feature map* (Rumelhart et al. 1988). Hence, the filter is analagous to a kernel convolution over the input data. Multiple feature detections require multiple filters and consequently multiple feature maps. A feature map is given by:

$$f^{(k)} = \sigma\Big(\sum_{i=0}^{m} \mathrm{w}^{(k)} \circ \mathrm{x}_i\Big), \qquad (1)$$

with m denoting the number of stars in the data set and k being the feature map index. Additionally, $w_0 = b$, and $x_0 = 1$ forms the input bias. The feature map can be said to represent a detected local spatial feature of the data. However, in image recognition, as a greater number of convolutional layers are added, the outputs of deeper layers often become increasingly difficult to interpret from a human visual perspective (Zeiler & Fergus 2013).

A pooling layer is commonly applied after convolutional layers. Pooling reduces the length of convolutional layer outputs by applying a comparison function over adjacent nodes in the layer outputs. In principle, this achieves a form of local spatial invariance within layers (Bengio 2013). Our pooling layers use a 4-node *max-pooling*, which condenses each adjacent 4 nodes into 1 node by selecting the maximum between them.



(a) An elementary input-output connection in a neural network layer. Each neuron (circle) holds a real number. The activation function, σ, maps the dot product of the weight vector, $\mathbf{w}$, and the input vector, $\mathbf{x}$, into a non-linear output.

(b) Three fully-connected neural network layers with one hidden layer and two output neurons.

(c) Convolutional filtering applied to a one-dimensional input. A kernel with weights (w_1, w_2, w_3) is moved over input neurons, 3 at a time. The weights across inputs here are fixed as a generic set of values per feature map, whereas in (a), each weight may be unique.

Figure 3. A general schematic of neural network layers.

2.4 Learning and Optimization

The objective of the convolutional neural network is to learn a particular set of weights from a training set that minimizes the error in approximating a ground truth y with a predicted output $\hat{y}$. In binary classification, y are binary values while $\hat{y}$ are 2-element vectors with output scores for each class. We use the softmax function at the output layer, such that each output node contains the value:

$$p(y = j|\mathbf{x}) = \frac{e^{\mathbf{x}\cdot\mathbf{w}_j}}{\sum_{k=1}^{K} e^{\mathbf{x}\cdot\mathbf{w}_k}}, \quad (2)$$

which defines the score of class j out of $K = 2$ classes. These scores are similar to probabilities, however, probability calibration is usually required to express these scores in terms of actual population probabilities. Since such work is beyond the scope of this Letter, it is sufficient to interpret the output scores as the prediction *likelihood* of a certain population class for a star. A score close to 1 indicates a high predicted likelihood for a HeB star, while a score close to 0 implies a high likelihood for an RGB star. We use a simple yet sufficient score threshold of 0.5 to assign the dominant population label of a target, such that a predicted score close to 0.5 implies a lack of classifier confidence in identifying the star's population label.

A suitable error function to minimize is the *cross-entropy* or log-loss (Murphy 2012), related to the difference between a true distribution, y, with a predicted distribution, $\hat{y}$. The minimization uses a mini-batch gradient descent algorithm, where the error derivatives with respect to a layer's weights are calculated from the end output and backpropagated to previous layers (Rumelhart et al. 1986). Layer weights are then updated accordingly over mini-batches (random subsets) of the training examples, which speeds up learning compared to updating over the full training set because it approximates the gradient and the error surface curvature (LeCun et al. 1998). We use a mini-batch size of 128 to train our network. During the training process, multiple passes of feed-forward and backpropagation are iterated until the error converges to a minimum. By training a convolutional neural network with a suitable optimization objective, we obtain a *classifier* that takes in our image as input and outputs a two-element vector $\hat{y}$ with the elements as the RGB and HeB likelihoods, respectively.

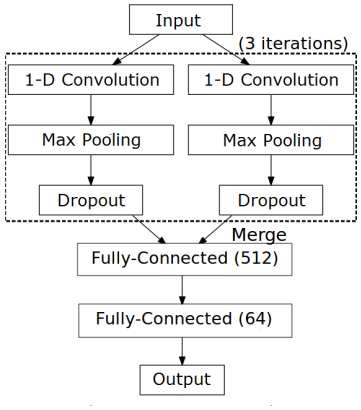


Figure 4. Structure of the convolutional neural network. The number in brackets for fully-connected layers is the input array length. The structure outlined by the dashed box is run three times before merging by tensor concatenation.

2.5 Classifier Structure and Hyperparameters

A deep learning classifier usually has multiple stacks of neural network layers on top of one another as its structure. This structure contains a vast combination of free parameters (*hyperparameters*), which have to be empirically determined. Our classifier structure (Figure 4) uses two parallel yet identical convolutional layer stacks, where both see the same input and iterate 3 times before merging outputs into a fully-connected layer. We use a filter size of 32 in each convolutional layer with 2/4/8 feature maps for iterations 1/2/3. This structure is defined by choosing the simplest, computationally cheap combination of structure and hyperparameters that has the best metric performance on a *hold-out validation set*. This set is part of the initial training data that is now left out in training for classifier structure selection.

To prevent overfitting on training data, we use *dropout* layers (Hinton et al. 2012), which randomly sets layer outputs to zero with probability p. This prevents layer outputs from memorizing the training data. We performed dropouts with $p = 0.6$ after each convolutional layer and constrain the norm of the layer's weight

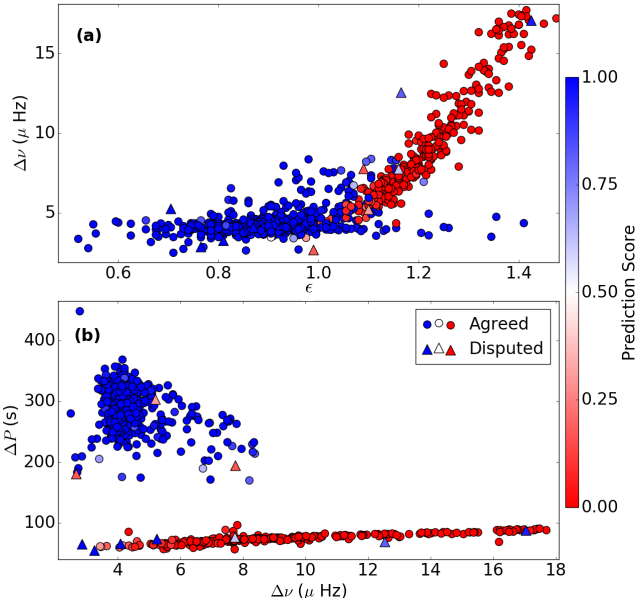


Figure 5. (a) $\Delta\nu - \epsilon$ diagram and (b) $\Delta P - \Delta\nu$ diagram of test set predictions. The colorbar corresponds to the score values of predictions, with deeper colors corresponding to a greater confidence towards a particular class (0 for RGB and 1 for HeB). Our classifications are available online.

vector to a value of 2. This combination has been shown to be effective at preventing overfitting (Srivastava et al. 2014). Models were constructed with the Keras library (Chollet 2015) built on top of Theano (Al-Rfou et al. 2016). With a Quadro K620 GPU, training requires approximately 1.5 hours with 200 training iterations, while predictions on thousands of new stars takes only a few seconds.

3 RESULTS

3.1 Classifier Performance

We report metrics on the mean of 10 separate hold-out validation sets (10-fold cross-validation) and on the test set. The metrics used to describe classifier perfomance are defined 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 $2\frac{P\times R}{P+R}$, with 1 as a perfect score.
ROC AUC: **R**eceiver **O**perating **C**haracteristic's **A**rea **U**nder **C**urve, which measures the classifier's average performance across all possible score thresholds. Has a value of 1 for a perfect classifier (Swets et al. 2000).
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.

Precison, recall, and the F1 scores complement accuracy in cases like ours where the population ratio is far from 50:50, while ROC AUC evaluates the overall classifier performance. Log loss reports the performance of class score outputs, with confident predictions rewarded low error when correct while penalised heavily otherwise. As observed in Table 1, the accuracy of the classifier on the cross-validation sets is generally above 98%. On the test set where the classifier benefits from training on the entire training set, the classifier is capable of classifying with a 99% accuracy and suffers a lower log loss. Having high values of precision, recall, and F1 score also indicates that the classifier is not heavily biased in predicting a particular population class that would not reflect the true population ratio.

Table 1. Metrics over the mean of 10-fold cross-validation (CV) and over the test set.

Dataset	CV (±1 std.)	Test
Accuracy	0.982 ± 0.005	0.990
Precision	0.982 ± 0.005	0.990
Recall	0.982 ± 0.005	0.991
F1 Score	0.982 ± 0.005	0.991
ROC AUC	0.998 ±0.002	0.996
Log Loss	0.055 ±0.020	0.044

Figure 5 shows the test set results in $\Delta\nu - \epsilon$ and $\Delta P - \Delta\nu$ diagrams. We derived the ϵ values using the method described in Stello et al. (2016a,b). One can see that 'disputed' predictions, namely predictions that are not in agreement with the "truth" labels from Vrard or Mosser, are more concentrated towards the low-$\Delta\nu$ regions. The classifier appears to be confident in most of its predictions (deep red and deep blue symbols), while most of the uncertain predictions are disputed. Upon inspection of the spectra of the 10 disputed stars, we visually verify that four of them, with 2.9μHz $< \Delta\nu < 5.2\mu$Hz, had incorrect ground truth labels. Another four are confirmed to be due to the classifier's inaccuracy. These stars have $\Delta\nu > 7.0\mu$Hz in the diagrams. The final two stars are "high" luminosity red giants with $\Delta\nu < 2.9\mu$Hz. Visual inspection was inconclusive as the spectrum of one had suppressed dipole modes with a moderate level of noise, while the other appeared much like an RGB star but was previously given a late HeB classification as its ground truth. From theory, we do not expect to see a clear difference between RGB and late HeB stars because of the lack of coupling between core and envelope in such stars (Stello et al. 2013, their Fig. 4b).

3.2 Classifying the Unclassified Set

We now use our trained classifier to predict the evolutionary state of the unclassified set (Figure 6). It can be seen that the predictions reflect the $\Delta\nu - \epsilon$ relation of RGB stars well (Huber et al. 2010) for the entire $\Delta\nu$ range spanned by the training set ($2.8\mu\text{Hz} \lesssim \Delta\nu \lesssim 18\mu\text{Hz}$). The secondary clump of HeB stars is seen in the diagram with $\Delta\nu \simeq 7 - 9\mu$Hz to the left of the RGB $\Delta\nu - \epsilon$ relation. In addition, the predictions also clearly show the HeB population at $\Delta\nu \simeq 3 - 4\mu$Hz, with ϵ values mostly ranging about 0.7 to 1.0. In Figures 5a and 6, stars with $\Delta\nu \simeq 4\mu$Hz and $\epsilon \gtrsim 1.2$ are low ϵ stars that have 'wrapped around' horizontally in the diagram.

Despite the predictions capturing the general distribution of red giant populations within the $\Delta\nu - \epsilon$ diagram , the classifier has its limitations from classifying based on image representation alone. For instance, it does not explicitly discriminate between frequency spacings, such that it can erronously predict HeB stars at high $\Delta\nu$ ($\Delta\nu \gtrsim 9\mu$Hz), where no HeB stars exist. However, only a very small fraction of predictions are subject to this inaccuracy. Another important limitation of these predictions is imposed by the

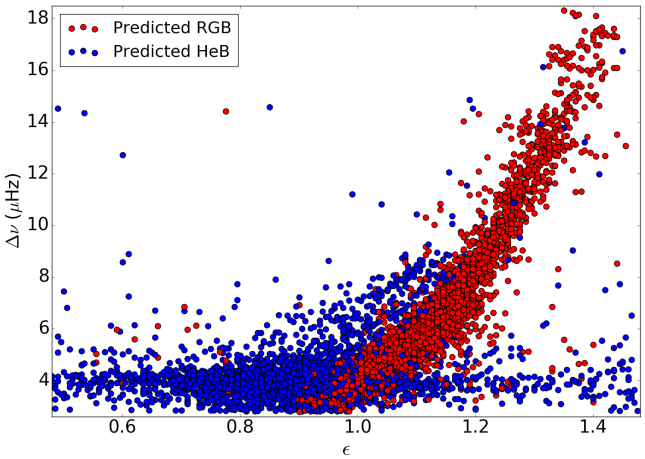


Figure 6. $\Delta\nu - \epsilon$ diagram of the unclassified set for stars with $\Delta\nu \gtrsim 2.8\mu$Hz. The classifications are available online.

parameter range of the training data. In principle, after learning the general 'image' of RGB stars, the predictions should extend beyond the range of $\Delta\nu$ values in training data. However, our classifier cannot yet do that. Our training data only includes RGB stars down to $\Delta\nu \simeq 2.8\mu$Hz, hence we infer that the reliability of the classifier predictions also holds to a similar $\Delta\nu$ threshold. Due to this, we do not provide classifications for 1139 red giants with $\Delta\nu < 2.8\mu$Hz in our unclassified set. We find that out of the 517 stars in our 'unclassified' set that have been previously classified by Stello et al. (2013), 43 of their predictions do not agree with ours. We visually verify that they had incorrectly labelled most of the disputed stars as HeB, with a majority having $\Delta\nu \simeq 5.5\mu$Hz. In addition, 1991 stars with $\Delta\nu \gtrsim 2.8\mu$Hz have been classified by Elsworth et al. (2016), from which our classifier produces disputing predictions for 195 stars. These disputes are mostly for those stars that they have predicted as RGB in the range of $2.8\mu\text{Hz} \lesssim \Delta\nu \lesssim 5\mu$Hz, which we find are split in roughly equal numbers into those that our classifier predicts correctly, those that our classifier predicts incorrectly, and those where we are uncertain of the true population by visual inspection. In the end, we produce new classifications for 5379 previously unclassified stars. In future work, we will develop methods to improve the generalization of classifier predictions across a greater range of asteroseismic parameters. In addition, we will also look into spectral features of high $\Delta\nu$ RGB stars, which can deceive the classifier to incorrectly predict the star as HeB.

4 CONCLUSIONS

We have developed a convolutional neural network to perform fast and efficient classification of red giant stars into those ascending the red giant branch and into those that have commenced core helium burning. We presented folded oscillation spectra as images, which contain visual features that are learned by the convolutional neural network. Training and testing on *Kepler* data yielded a 98% cross-validation accuracy and a 99% test set accuracy, benchmarked against classifications based on asymptotic period spacing measurements. Out of the predictions that were in conflict with the 'ground truth', most scenarios of classifier inaccuracy were limited to the intermediate to high $\Delta\nu$ range, whereas for several low $\Delta\nu$ disputed cases, the input population labels were either incorrect or ambiguous based on visual inspection.

We also made predictions on 7655 *Kepler* red giants that do not have their asymptotic period spacing measured, from which 5379 have not been previously classified by any other means. We observed good agreement with the expected distribution of red giant populations in $\epsilon - \Delta\nu$ space for $\Delta\nu > 2.8\mu$Hz. Despite being currently limited to predicting within the asteroseismic parameter ranges of the training set, this new, simple, and effective method of classifying oscillation spectra seems promising for further future classifications on large datasets in asteroseismology.

REFERENCES

Al-Rfou R., et al., 2016, arXiv e-prints, abs/1605.02688
Bedding T. R., et al., 2011, Nature, 471, 608
Bengio Y., 2013, preprint (arXiv:1305.0445)
Borucki W. J., et al., 2010, Science, 327, 977
Chollet F., 2015, Keras, https://github.com/fchollet/keras
Christensen-Dalsgaard J., Silva Aguirre V., Elsworth Y., Hekker S., 2014, MNRAS, 445, 3685
Dupret M.-A., et al., 2009, A&A, 506, 57
Elsworth Y., Hekker S., Basu S., Davies G., 2016, preprint (arXiv:1612.04751)
Fukushima K., 1980, Biological Cybernetics, 36, 193
Garcia C., Delakis M., 2004, IEEE Transactions on Pattern Analysis and Machine Intelligence, 26, 1408
Hinton G. E., Srivastava N., Krizhevsky A., Sutskever I., Salakhutdinov R. R., 2012, preprint (arXiv:1207.0580)
Huber D., Stello D., Bedding T. R., Chaplin W. J., Arentoft T., Quirion P.-O., Kjeldsen H., 2009, Communications in Asteroseismology, 160, 74
Huber D., et al., 2010, ApJ, 723, 1607
Kallinger T., et al., 2012, A&A, 541, A51
LeCun Y., Bengio Y., 1998, MIT Press, Cambridge, MA, USA, Chapt. Convolutional Networks for Images, Speech, and Time Series, pp 255–258
LeCun Y., Bottou L., Orr G. B., Müller K.-R., 1998, in Neural Networks: Tricks of the Trade. Springer-Verlag, London, UK, UK, pp 9–50
Mosser B., et al., 2014, A&A, 572, L5
Murphy K. P., 2012, Machine Learning. MIT Press Ltd
Nair V., Hinton G. E., 2010, in FÃijrnkranz J., Joachims T., eds, Proceedings of the 27th International Conference on Machine Learning (ICML-10). Omnipress, pp 807–814
Ricker G. R., et al., 2014, in Space Telescopes and Instrumentation 2014: Optical, Infrared, and Millimeter Wave. p. 914320 (arXiv:1406.0151)
Rosenblatt F., 1962, Principles of neurodynamics: perceptrons and the theory of brain mechanisms. Report (Cornell Aeronautical Laboratory), Spartan Books
Rumelhart D. E., Hinton G. E., Williams R. J., 1986, Nature, 323, 533
Rumelhart D. E., Hinton G. E., Williams R. J., 1988, MIT Press, Cambridge, MA, USA, Chapt. Learning Internal Representations by Error Propagation, pp 673–695
Sermanet P., Kavukcuoglu K., Chintala S., LeCun Y., 2012, preprint (arXiv:1212.0142)
Srivastava N., Hinton G., Krizhevsky A., Sutskever I., Salakhutdinov R., 2014, J. Mach. Learn. Res., 15, 1929
Stello D., et al., 2013, ApJ, 765, L41
Stello D., Cantiello M., Fuller J., Garcia R. A., Huber D., 2016a, Publ. Astron. Soc. Australia, 33, e011
Stello D., Cantiello M., Fuller J., Huber D., García R. A., Bedding T. R., Bildsten L., Silva Aguirre V., 2016b, Nature, 529, 364
Swets J. A., Dawes R. M., Monahan J., 2000, Scientific American, pp 82–87
Vrard M., Mosser B., Samadi R., 2016, A&A, 588, A87
Zeiler M. D., Fergus R., 2013, preprint (arXiv:1311.2901)

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