Size does matter: Exoplanet detection with a sparse convolutional neural network

Visser, K.; Bosma, B.; Postma, E.

published in
Astronomy and Computing
2022

DOI (link to publisher)
10.1016/j.ascom.2022.100654

document version
Publisher's PDF, also known as Version of record

citation for published version (APA)
Visser, K., Bosma, B., & Postma, E. (2022). Size does matter: Exoplanet detection with a sparse convolutional neural network. *Astronomy and Computing*, *41*, 1-6. Article 100654. Advance online publication. https://doi.org/10.1016/j.ascom.2022.100654

Astronomy and Computing 41 (2022) 100654

Contents lists available at ScienceDirect

Astronomy and Computing

journal homepage: www.elsevier.com/locate/ascom

Full length article

Size does matter: Exoplanet detection with a sparse convolutional neural network

K. Visser [a,*], B. Bosma [b], E. Postma [a,c]

[a] Jheronimus Academy of Data Science, Sint Janssingel 92, 5211 DA, 's-Hertogenbosch, The Netherlands
[b] Vrije Universiteit Amsterdam, De Boelelaan 1105, 1081 HV, Amsterdam, The Netherlands
[c] CSAI, Tilburg University, Warandelaan 2, 5037 AB, Tilburg, The Netherlands

ARTICLE INFO

Article history:
Received 25 June 2022
Accepted 14 September 2022
Available online 21 September 2022

Keywords:
Exoplanet detection
Convolutional neural networks

ABSTRACT

In recent years, convolutional neural networks (CNNs) have been successfully applied to the task of exoplanet detection. The success of CNNs in many domains has been attributed to overparameterization. In the domain of image recognition, larger CNNs with more free parameters tend to result in better predictions. In exoplanet detection, the reverse seems to apply. Astronet and Exonet, the first CNNs for exoplanet detection had more than 10 million parameters. Exonet-XS is a much smaller network (0.85 million parameters) that uses domain knowledge and performs almost on par with its larger variant Exonet. In earlier work we proposed Genesis, an even smaller network with less than half of the parameters of Exonet-XS (0.39 million). In this paper we examine and compare the performances of Genesis, Exonet-XS, and Exonet (all of which include stellar parameters as domain knowledge) using a proper cross-validation procedure. We find that Genesis, despite its smaller size, outperforms Exonet-XS and performs at a level comparable with Exonet. Hence, reducing overparameterization seems to be beneficial to exoplanet detection from folded light curves. In an attempt to uncover the cause of Genesis' success, we study the effect of varying the hyperparameters controlling the convolutional and pooling layers. We find that the performance of Genesis can be explained by its sparsity and use of large convolution filters and pooling strides. We conclude that Genesis and the use of large filters and pooling strides are to be preferred for exoplanet detection over previously-proposed architectures.

1. Introduction

CNNs have been shown to excel in the detection of exoplanets from transit data (Ansdell et al., 2018; Valizadegan et al., 2022; Shallue and Vanderburg, 2018; Yu et al., 2019; Schanche et al., 2019). Specifically, Shallue and Vanderburg (2018) show that Astronet outperforms logistic regression on the task of exoplanet detection from folded light curves. Astronet achieved a detection accuracy of 96%, while logistic regression yielded only 91.7%. Ansdell et al. (2018) developed an extension of Astronet, called Exonet, that added domain knowledge (centroid time series and stellar parameters) to suppress false detections. This improved the detection accuracy to 97.5%. Both Astronet and Exonet, however, consist of an excessive number of free parameters. Whilst overparameterization has been shown to be beneficial to the performance of deep learning models in domains such as vision and language (Szegedy et al., 2015; He et al., 2016), exoplanet detection from (folded) light curves may be less complex. Hence, Ansdell et al. (2018) also proposed a much smaller variant of Exonet, called Exonet-XS, consisting of 0.85 million parameters (a reduction of 92% with respect to Exonet). This far sparser architecture performed only slightly worse than Exonet (detection accuracy is 96.6%, 0.9 percent point lower than the accuracy of Exonet).

In previous work, we presented Genesis (Visser et al., 2022), an even sparser architecture than Exonet-XS. We performed a careful comparative evaluation, using a cross-validation procedure, to obtain performance distributions rather than point estimates. We found that Genesis performed on a par with Exonet-XS, suggesting that a further sparsification of the architecture is feasible. However, we did not include domain knowledge (centroid time series and stellar parameters) in our implementations of Exonet-XS and Genesis. In this paper, we perform a proper comparative evaluation of Genesis, Exonet-XS, and Exonet that includes domain knowledge in all three architectures. We find that Genesis outperforms Exonet-XS and performs on par with Exonet. This indicates that with domain knowledge included, the sparse architecture of Genesis is to be preferred over Exonet and Exonet-XS. In addition, we examine what architectural characteristics are responsible for the success of Genesis.

The remainder of this paper is organized as follows. In Section 2 we present the architectural compositions of Genesis,

* Corresponding author.
E-mail address: k.visser@tue.nl (K. Visser).

https://doi.org/10.1016/j.ascom.2022.100654
2213-1337/

Table 1
Overview of the main characteristics of Genesis, Exonet-XS, and Exonet. The entries with a '+' indicate the values for the two branches. Exonet-XS and Exonet have been slightly altered from the original versions presented by Ansdell et al. (2018) to use SOFTMAX-2 instead of SIGMOID).

	Genesis	Exonet-XS	Exonet
# parameters (millions)	0.39	0.85	10.50
# views	1	2	2
# centroids	yes	yes	yes
# convolution layers	4	3 + 2	10 + 4
# convolution stride	1	1 + 1	1 + 1
# pooling layers	1	2 + 1	4 + 2
# pooling stride	80	2 + 2	2 + 2
# Stellar data	yes	yes	yes
# dense layers	2	1	4
# neurons	256	512	512
Last pooling layer	global average pool	global max pool	global max pool
Dropout	yes (25%)	no	no
Output transfer function	SOFTMAX-2	SOFTMAX-2	SOFTMAX-2

Exonet-XS, and Exonet. Thereafter, Section 3 outlines the experimental procedure used to compare the three architectures. The results of our comparative evaluation are presented in Section 4.

2. Architecture

Table 1 lists the characteristics of the three architectures central to our study. The first row, listing the number of free parameters in millions, shows that Exonet-XS has about 8% of the parameters of Exonet and Genesis less than 4%. The second row reveals the main architectural difference between Genesis and both Exonet architectures. Where the latter have two input views, one processing a global view and one a local view of the folded light curve, Genesis relies on a single input view (Visser et al., 2022). The use of a single view leads to a reduced number of convolution layers (4 instead of 14 and 5 for Exonet and Exonet-XS, respectively). Furthermore, Genesis relies on two dense layers with 256 neurons each, whereas Exonet-XS has one dense layer with 512 neurons. The final two distinguishing features of Genesis are the last pooling layer that uses average pooling, rather than global max pooling used in the Exonet architectures, and the use of dropout that is not used in the Exonet architectures.

Fig. 1 visually represents the three architectures. All inputs consist of (folded) light curves and centroid time series. After a sequence of convolution and pooling layers, all three networks combine the processed inputs with stellar parameters. The combination of processed inputs and stellar parameters feed the results into one or more fully connected (dense) layers and a softmax output.

3. Experimental procedure

Genesis, Exonet-XS, and Exonet were implemented in Python using the Keras library for deep learning (Chollet et al., 2015).

3.1. Data

For our experiments we made use of the data set used by Ansdell et al. (2018) that followed the procedure of Shallue and Vanderburg (2018) (see Visser et al., 2022 for further details). The data set is based on the Q1–Q17 Data Release 24 (DR24) light curves from the Mikulski Archive for Space Telescopes (Thompson et al., 2015) and consists of 15,084 light curves, i.e., long cadence brightness values obtained by fixed-aperture photometry by summing over the pixel values within the aperture (Twicken et al., 2010). The light curves contain one or more potential exoplanet transits, or "threshold crossing events" (TCEs). Only a minority of the TCEs represents exoplanet transits, the majority is due to eclipsing binaries, instrumental noise, or spurious events. Of the 15,084 TCEs, 3454 were labeled as positive (exoplanet candidate) and 11,630 as negative (no exoplanet) (McCauliff et al., 2015). All light curves are flattened using iterative spline fitting where outliers (values $> 3\sigma$) are removed by a variant of sigma clipping. Using binning, a local view (201 bins) and a global view (2001 bins) are created for each phase-folded TCE. Both views are normalized to the interval $[0, -1]$, with a value of -1 representing the maximum transit depth.

In addition to the light curves, the data set of Ansdell et al. (2018) contains time series of the pixel positions of the center of light within an aperture (centroids). These centroid time series help to distinguish exoplanet transits from eclipsing-binary transits. In case of a single star, the displacement is expected to be zero. In case of an eclipsing binary, the displacement will be non-zero (apart from the moment the secondary is transiting), because the companion star 'pulls' the centroid away from the center.

Finally, Ansdell et al. (2018) included stellar parameters to their data set. Specifically, the stellar effective temperature, surface gravity, metallicity, radius, mass, and density. Each of these parameters was normalized across the training set by subtracting the median value and dividing by the standard deviation.

3.1.1. Data partitioning

Earlier evaluations of Astronet (Shallue and Vanderburg, 2018) and the Exonet architectures (Ansdell et al., 2018) considered detection performance within a single fixed partitioning of the data set into training, validation, and test sets. Such a partitioning results in point estimates of the accuracy and AUC, which hampers a proper comparative evaluation. Hence, as in our previous work (Visser et al., 2022), we employ Monte Carlo Cross Validation (MCCV). Following this procedure, training and evaluation of an architecture is repeated n times (in our case, $n = 20$). Each experiment is performed with randomly resampled training, validation, and test sets consisting of 80%, 10%, and 10% of the TCEs, respectively. The MCCV procedure mitigates the effect of the assignment of TCEs to either set and yields a distribution for performance metrics instead of point estimates.

3.1.2. Data augmentation

We applied data augmentation by adding horizontal reflections of the phase-folded light curves (Shallue and Vanderburg, 2018), as well as light curves corrupted by Gaussian noise (Ansdell et al., 2018). For the Exonet architectures, Gaussian noise is added to both the global and local view. The additive noise

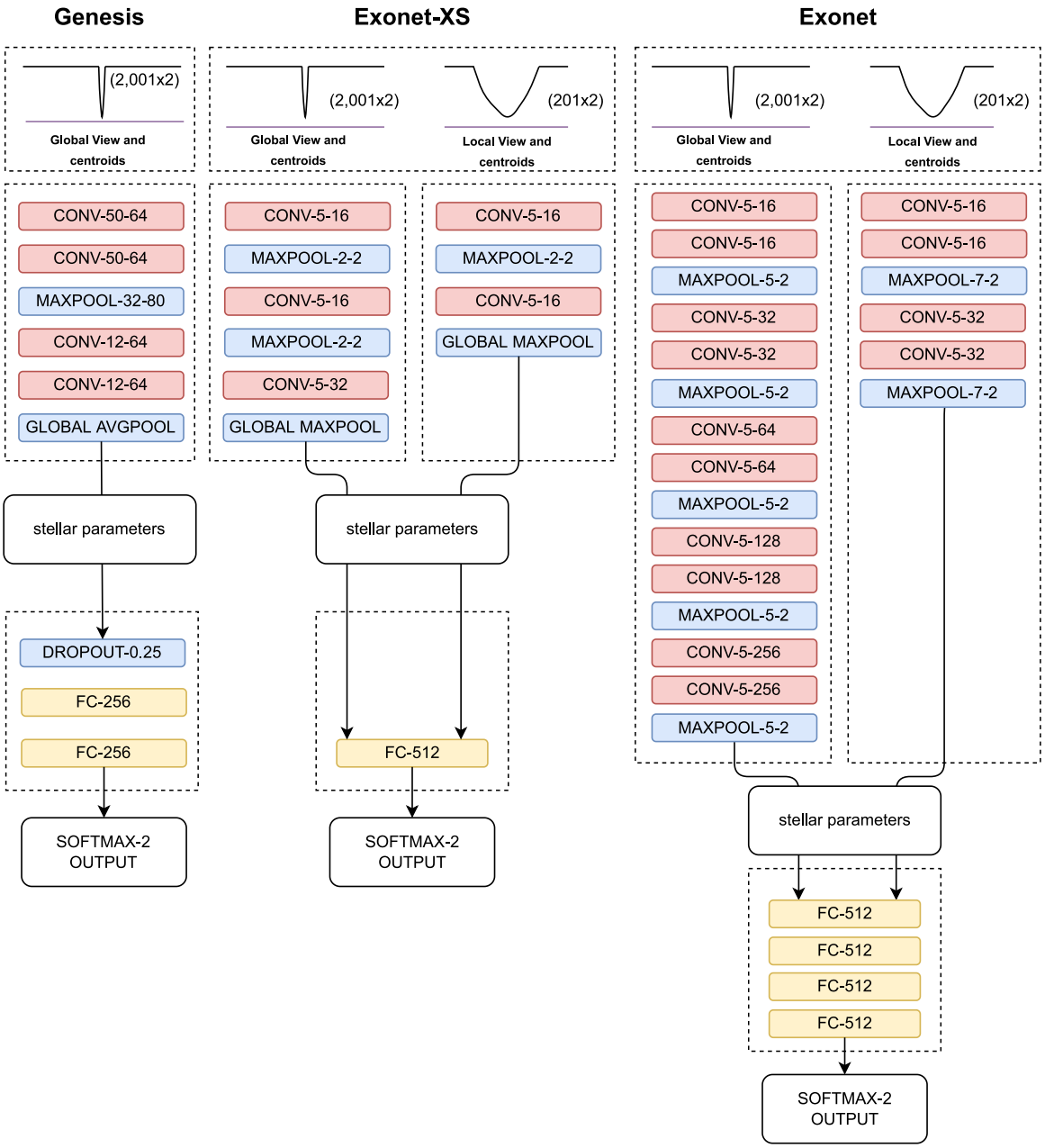

Fig. 1. Schematic representations of the architectures implemented by us in Keras. The three key characteristics of Genesis are: (1) input of single-view phase folded light curves with binsize 2001 instead of dual-view input with binsize 2001 and 201, (2) reduced number of convolution and dense layers (8 in Genesis, 11 in Exonet-XS, and 25 in Exonet), and (3) dropout layer. Explanation of the abbreviations: CONV-a-b represents a convolutional layer of size a with b filters; MAXPOOL-a-b indicates the pooling layer with length a and stride b; AVGPOOL-a stands for average pooling over a filter outputs of the preceding layer; DROPOUT-a is a dropout layer with a dropout probability of a%; FC-a represents a fully connected layer with a hidden neurons each. Stellar parameters help to reduce the False Positive Rate caused by Binary Eclipsing Binaries (BEBs). Domain knowledge consists of *centroids* and *stellar parameters*.

is sampled from a normal distribution with mean zero and a standard deviation that is sampled from a uniform distribution over the interval [0, 1]. As shown by Ansdell et al. (2018), the addition of Gaussian noise helps to improve the precision and recall capabilities of Exonet.

3.2. Experimental settings

All experiments were performed with the same settings discussed in Visser et al. (2022). Each model instance is trained for a maximum of 125 epochs, with a batch size of 1024. As an output transfer function, SOFTMAX-2 is used with categorical cross-entropy as the loss function. For all other transfer functions we employed ReLUs. Training is performed with the Adam optimizer (Kingma and Ba, 2015) with learning rate $\alpha = 0.001$, exponential decay rates $\beta_1 = 0.9$, $\beta_2 = 0.999$, and $\hat{\varepsilon} = 10^{-7}$.[1] We use early stopping with `min_delta` = 0.1% and `patience` = 50 epochs.

3.3. Evaluation

In the comparative evaluation of the three architectures, we perform for each architecture $n = 20$ independent replications of the model averaging procedure introduced by Shallue and

[1] In Keras $\hat{\varepsilon}$ is used instead of ε for computational efficiency.

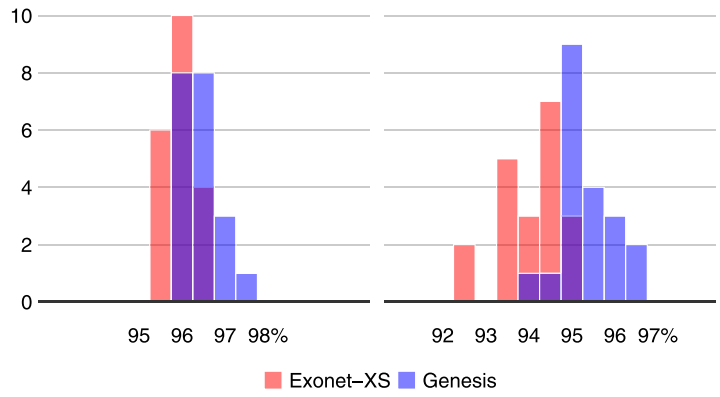

Fig. 2. Genesis and Exonet-XS. Left: Mean ACC histograms. Right: Mean AUC histograms.

Table 2
Overview of the ensembled performances of Genesis, Exonet-XS, and Exonet. The standard deviations are given in parentheses.

	Mean ACC	Mean AUC
Genesis	96.4% (0.4%)	95.3% (0.6%)
Exonet-XS	96.0% (0.4%)	94.1% (0.7%)
Exonet	96.8% (0.6%)	95.6% (0.9%)

Vanderburg (2018). Summarizing the resulting distribution, we define Mean ACC and Mean AUC as the average ACC and AUC values over the $n = 20$ independent replications.

4. Results

Table 2 presents the results of our experiments. For the three architectures examined, the Mean ACC and Mean AUC are listed. For both metrics, the standard deviation is given in parentheses.

We make the following observations. The inclusion of domain knowledge (stellar parameters and folded centroid time series) results in a higher Mean ACC and Mean AUC for Genesis compared to Exonet-XS (96.4% versus 96.0%, and 95.3% versus 94.1%, respectively).

Moreover, we notice that the standard deviations of Genesis and Exonet-XS over the ensembles are similar, meaning that the level of variance over the randomly resampled data of the MCCV procedure is comparable. Fig. 2 illustrates the distributions of Mean ACC (left) and Mean AUC (right) over the resampled data. Although the distributions for Genesis and Exonet-XS overlap, the former clearly has a larger mean value.

Comparing the performances of Genesis and Exonet, we observe that Exonet has a slightly higher Mean ACC and Mean AUC. However, given the standard deviations, this difference is not significant. In other words, Exonet and Genesis perform on a par. Fig. 3 illustrates the large degree of overlap of the Mean ACC and Mean AUC distributions.

5. Discussion

Our results reveal that reducing the number of parameters does not or barely affect exoplanet detection performance. Apart from a reduction of parameters, the architecture of Genesis has four main characteristics.

The first characteristic of the Genesis architecture is the inclusion of a dropout layer that is not included in both Exonet architectures. Second, the architecture of Genesis is similar to that of Exonet with respect to the use of pairs of convolution layers that alternate with pooling layers. In contrast, Exonet-XS uses single convolution layers that alternate with pooling layers. Third, the convolution layers of Genesis have a larger size and more filters than those in the Exonet-XS architecture. Whereas the sizes of the convolution layers in both Exonet architectures remain constant, for Exonet the number of filters increases with depth for its global-view branch. Fourth, the stride of the pooling layer of Genesis is much larger than that of Exonet-XS and Exonet. In fact, the stride results in a degree of skipping of convolution outputs of the preceding layer, leaving some gaps of unprocessed convolution outputs. Denoting the second convolution-layer elements by $c_1, c_2, \ldots$, the first pooling is performed over c_1 to c_{32}, and the second pooling over c_{81} to c_{101}, due to the stride of 80. This means that elements c_{33} to c_{80} (and those associated with the other gaps) are not subject to any pooling or further processing. Apparently, the cascade of filters in the two initial convolution layers that feed into the pooling layer, collect sufficient information from the input to enable successful detection performance. This would imply that even Genesis is (slightly) overparameterized and that a further reduction of parameters is feasible. The use of large pooling strides is highly uncommon, but can be effective (Wu and Perin, 2021).

The first characteristic of Genesis is generally known to benefit performance (Goodfellow et al., 2016). The second characteristic, shared with Exonet, may also be partially responsible for Genesis' performance. To determine the contribution of the first, third, and fourth characteristics, we performed additional experiments to see how varying the associated hyperparameter values affect detection performance. More specifically, we studied the effect of excluding dropout, reducing the size of the convolution filters, and reducing the stride of the pooling layers. We also examined the effect of the combination of the latter two.

In doing so, we performed full training and testing using the same data set and experimental settings as specified above. Table 3 lists the results. The top two rows show the results and hyperparameters of Genesis with and without dropout. The next two rows show the results for reduced sizes of the first and second pair of convolution filters (*Conv1* and *Conv2*, respectively). As can be seen, the Mean ACC and Mean AUC values are lower for these model specifications, which indicates that the large convolution filter size (partially) contributes to the performance of Genesis. Similarly, the lower Mean ACC for the row labeled *Reduced pooling stride* suggests that a smaller stride for the pooling layer negatively affects the performance of Genesis. This is also the case for the stride and filter size for max pooling used in Exonet, as can be seen in the row labeled *Reduced pooling stride (Exonet)*. Finally, combining both (rows labeled *Combined*) shows a similar effect. Taken together, these findings suggest that, apart from an overall sparser architecture, the large convolution filter sizes and large pooling stride, contributed to the performance of Genesis.

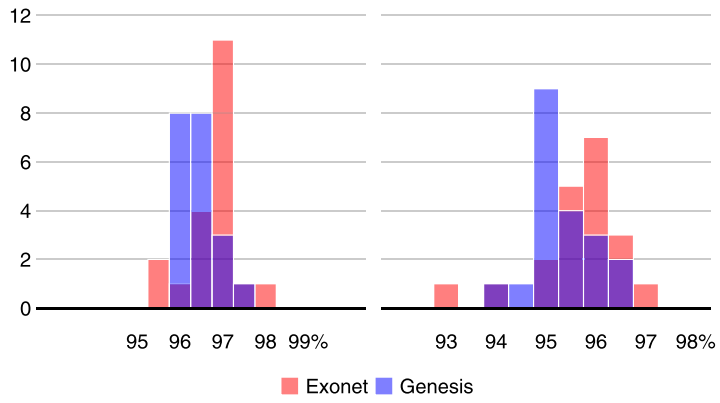

Fig. 3. Genesis and Exonet. Left: Mean ACC histograms. Right: Mean AUC histograms.

Table 3
The effect on Mean ACC and Mean AUC of adjusting model hyperparameters of Genesis. In bold are the best mean performances and corresponding standard deviations (in parentheses) for each of the classes of adjustments made. In the last three columns, the underlined numbers indicate the hyperparameter values that are changed with respect to the original Genesis architecture.

Model specification	Mean ACC	Mean AUC	Conv1	Conv2	Maxpool
Original Genesis architecture	**96.4**% (0.4%)	**95.3**% (0.6%)	50–64	12–64	32–80
Genesis architecture without dropout	96.3% (0.4%)	95.2% (0.7%)	50–64	12–64	32–80
Reduced sizes convolution filters	**96.3**% (0.4%)	**94.9**% (0.7%)	25–64	6–64	32–80
Reduced sizes convolution filters	95.8% (0.4%)	94.2% (0.6%)	12–64	3–64	32–80
Reduced pooling stride	**96.2**% (0.4%)	**94.9**% (0.7%)	50-64	12–64	32–40
Reduced pooling stride (Exonet)	96.2% (0.4%)	94.7% (0.6%)	50–64	12–64	7–2
Combined	**96.3**% (0.4%)	**94.9**% (0.7%)	25–64	6–64	32–40
Combined	96.2% (0.4%)	94.7% (0.6%)	12–64	3–64	32–40

6. Conclusion and future work

We conclude that sparse architectures such as Genesis, and the use of a single input view and large convolution filters and pooling strides are to be preferred for exoplanet detection over previously proposed architectures with two input views, small convolution filters and small pooling strides. Our future work will use our insights in an attempt to improve the performance of the recently-proposed Exominer architecture (Valizadegan et al., 2022).

CRediT authorship contribution statement

K. Visser: Conceptualization, Methodology, Software, Validation, Formal analysis, Investigation, Resources, Data curation, Writing – original draft, Writing – review & editing, Visualization, Project administration. **B. Bosma:** Conceptualization, Methodology, Validation, Formal analysis, Writing – original draft, Writing – review & editing, Visualization, Supervision, Project administration. **E. Postma:** Conceptualization, Methodology, Validation, Formal analysis, Writing – original draft, Writing – review & editing, Supervision, Project administration.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Data availability

Data will be made available on request.

References

Ansdell, M., Ioannou, Y., Osborn, H., 2018. Scientific domain knowledge improves exoplanet transit classification with deep learning. Astrophys. J. Lett. 869 (1), L7.

Chollet, F., et al., 2015. Keras. https://keras.io.

Goodfellow, I., Bengio, Y., Courville, A., 2016. Deep Learning. MIT Press.

He, K., Zhang, X., Ren, S., Sun, J., 2016. Deep residual learning for image recognition. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. CVPR, pp. 770–778.

Kingma, D., Ba, J., 2015. Adam: A method for stochastic optimization. In: Proceedings of the 3rd International Conference on Learning Representations. ICLR 2015.

McCauliff, D., Jenkins, J., Catanzarite, J., 2015. Automatic classification of Kepler planetary transit candidates. Astrophys. J. 806 (1), 6.

Schanche, N., Cameron, A.C., Hébrard, G., Nielsen, L., Triaud, A., Almenara, J., Alsubai, K., Anderson, D., Armstrong, D., Barros, S., 2019. Machine-learning approaches to exoplanet transit detection and candidate validation in wide-field ground-based surveys. Mon. Not. R. Astron. Soc. 483 (4), 5534–5547.

Shallue, C., Vanderburg, A., 2018. Identifying exoplanets with deep learning: A five-planet resonant chain around Kepler-80 and an eighth planet around Kepler-90. Astron. J. 155 (2), 94.

Szegedy, C., Liu, W., Jia, Y., Sermanet, P., Reed, S., Anguelov, D., Erhan, D., Vanhoucke, V., Rabinovich, A., 2015. Going deeper with convolutions. In: 2015 IEEE Conference on Computer Vision and Pattern Recognition. CVPR, pp. 1–9.

Thompson, S., Jenkins, J., Caldwell, D., 2015. Kepler Data Release 24 Notes (KSCI-19064-002). NASA Ames Research Center.

Twicken, J., Clarke, B., Bryson, S., Tenenbaum, P., Wu, H., Jenkins, J., Girouard, F., Klaus, T., 2010. Photometric analysis in the Kepler science operations center pipeline. In: Proceedings Volume 7740, Software and Cyberinfrastructure for Astronomy.

Valizadegan, H., Martinho, M., Wilkens, L., Jenkins, J., Smith, J., Caldwell, A., Twicken, J., Gerum, P., Walia, N., Hausknecht, K., Lubin, N., Bryson, S., Oza, N., 2022. ExoMiner: A highly accurate and explainable deep learning classifier that validates 301 new exoplanets. Astrophys. J. 926 (2), 120.

Visser, K., Bosma, B., Postma, E., 2022. Exoplanet detection with genesis. J. Astron. Instrum. 11 (3), 7.

Wu, L., Perin, G., 2021. On the importance of pooling layer tuning for profiling side-channel analysis. In: Applied Cryptography and Network Security Workshops. Springer International Publishing, pp. 114–132.

Yu, L., Vanderburg, A., Huang, C., Shallue, C., Crossfield, I., Gaudi, B., Daylan, T., Dattilo, A., Armstrong, D., Ricker, G., Vanderspek, R., Latham, D., Seager, S., Dittmann, J., Doty, J., Glidden, A., Quinn, S., 2019. Identifying exoplanets with deep learning. III. Automated triage and Vetting of TESS candidates. Astron. J. 158 (1), 25.