SF2A 2018
P. Di Matteo, F. Billebaud, F. Herpin, N. Lagarde, J.-B. Marquette, A. Robin, O. Venot (eds)

DEEP LEARNING DETERMINATION OF STELLAR ATMOSPHERIC FUNDAMENTAL PARAMETERS

R. Kou[1], P. Petit[1], F. Paletou[1], L. Kulenthirarajah[1] and J.-M. Glorian[1]

Abstract. In order to estimate fundamental parameters (effective temperature, surface gravity and metallicity) of the large amount of stars in the PolarBase data base, we need a fast and reliable algorithm. With this aim, we developed a convolutional neural network able to derive this parameter triplet. Our neural network was trained on observed spectra from the PolarBase and Elodie data bases (M to F stars). We used the spectral region between 6095 and 6185 Angströms which has proved its efficiency in a number of previous studies. We analyzed the outcome of our approach for a sample of spectra from the same data bases. We discuss the accuracy and reliability of the neural network depending on the parameter domain, size and quality of the training data set.

Keywords: stars, sun-like stars

1 Introduction

High resolution stellar spectra contain a wealth of information about fundamental parameters of the targeted objects (effective temperature, surface gravity, metallicity). The accurate determination of stellar physical quantities is critical for a number of subsequent studies (e.g. for the measurement of planetary masses obtained through velocimetric methods).

We propose here to train a convolutional network to recover stellar parameters throughout a wide domain, with spectral types from M to F, and surface gravities from supergiants to main sequence dwarfs. Our approach is directly inspired from the work of Fabbro et al. (2018). We first describe the architecture of our convolutional network, then detail the training data set and finally discuss the outcome for our validation data set.

2 Architecture of the neural network

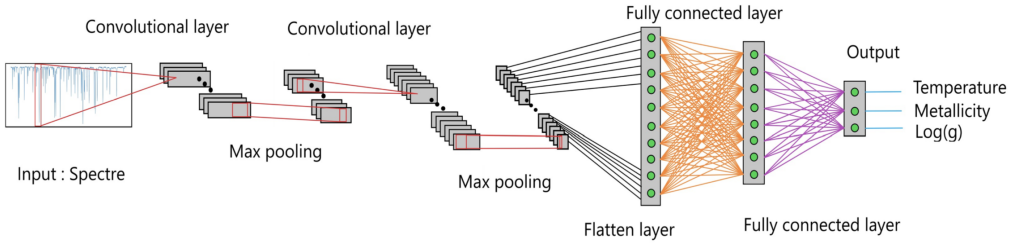

Fig. 1. Structure of the neural network used to recover stellar fundamental parameters.

The convolutional neural network (CNN) is composed of several layers (Fig. 1). The spectrum is used as an input vector whereas the output is constituted of the selected fundamental parameters (here the effective temperature, surface gravity, and metallicity). Here, the spectral domain was restricted to the region between

[1] IRAP, Université de Toulouse, CNRS, UPS, CNES, 31400, Toulouse, France

6095 and 6185 Angströms, as successfully done in a number of previous studies (e.g. Valenti & Fischer 2005; Brewer et al. 2015).

At first, one or several convolutional layers are used in order to learn to recognize the local information from the spectrum, *i.e.* the shapes of spectral lines. For each convolutional layer, we have to set the size of the sliding windows which will convolve the spectrum, and the number n of filters we want to use, so that the output of the convolutional layer will be constituted of n vectors.

After each convolutional layer, we insert a max pooling layer whose aim is to reduce the number of points produced by the convolutional layers. The max pooling layer only keeps the most significant points, so that only the interesting information is kept for further analysis. This layer helps to accelerate the training process because, by removing the points that do not bring any useful information, the number of weights used in the following layers can be reduced.

Following this step, a flatten layer enables to concatenate the vectors in input of this layer in order to produce a unique vector, to be used as an input for the last type of layers we use, which is the fully connected layer. This layer makes linear combinations between the points produced by the previous layer and some weights. Since the points produced by the fully connected layer arise from linear combinations of all the input points, this step enables to mix the information coming from everywhere in the spectrum. Finally, we use a last fully connected layer in order to produce three values that will correspond to temperature, metallicity and $\log(g)$.

Of course, at the beginning of the training process there is no reason for the three generated values to be the right parameters. This is why a training phase is required. We use a training set composed of spectra for which we already know the values of the output parameters. During the training, the CNN tries to find the parameters and measures the resulting error thanks to a loss function. The weights (in the convolutional and fully connected layers) of the CNN are then updated in order to minimize this loss function. Once trained, the CNN can be used to predict the parameters of stars for which they are not already known.

3 Training data set

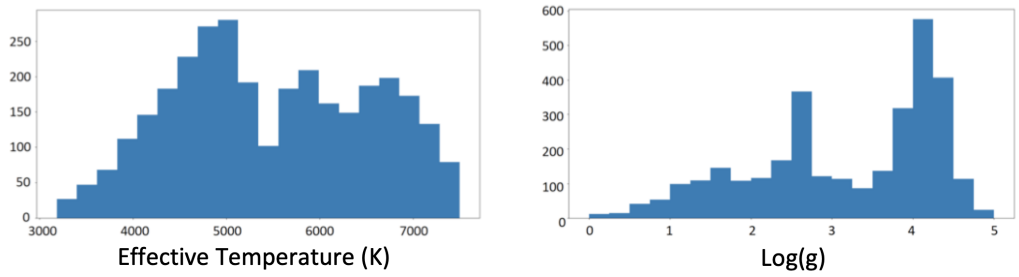

Fig. 2. Left: Distribution of effective temperatures in the training set. **Right:** Same as the left panel, for surface gravity.

Spectra used as our training set are issued from the PolarBase and Elodie data bases (Petit et al. 2014; Moultaka et al. 2004). 1332 stars have been considered, for a total of 3129 spectra (up to 5 spectra of a given star were allowed to be selected). Their fundamental parameters were recovered through an automated query of the VizieR data base (Paletou & Zolotukhin 2014), and the median of each returned parameter was selected for stars with multiple values of a given parameter listed in VizieR. The resulting range in effective temperature goes from 3182 K to 7500 K, while $\log(g)$ goes from 0 to 5, and $[Fe/H]$ from -0.7 to +0.7 (Fig. 2).

4 Results

The validation data set was constituted of 196 stars, again taken from the PolarBase and Elodie archives. With up to five spectra per star, we end up with a total of 379 spectra for which fundamental parameters can be obtained both from VizieR and from our neural network. The outcome for the effective temperature is illustrated in Fig. 3, where a good correspondence is observed between the two sets of values, with a spread that tends to increase with the temperature. No significant bias is observed, except in the upper right of the plot where reconstructed temperatures are systematically lower than the VizieR values.

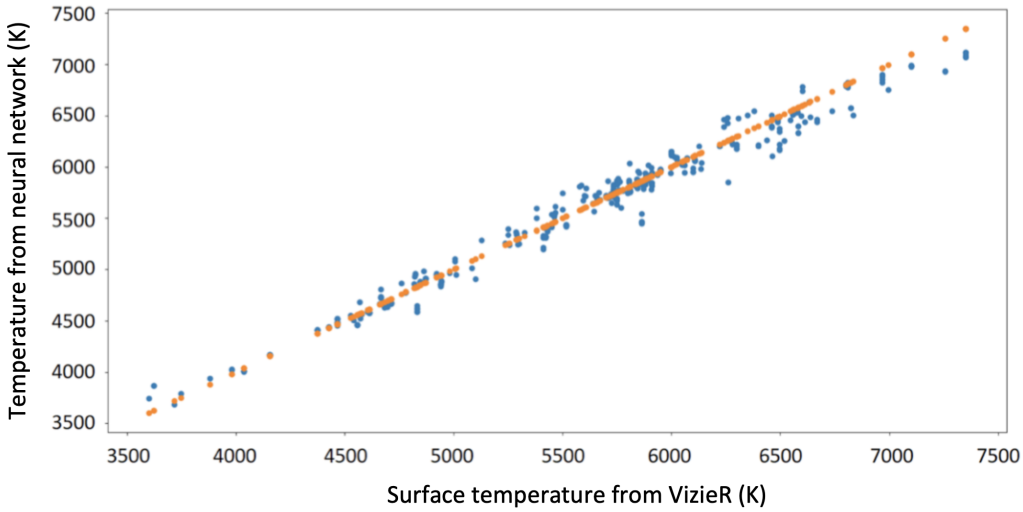

Fig. 3. Effective temperatures reconstructed by the neural network (Y axis), compared to the effective temperature given by VizieR (X axis). A perfect tool would have produced the orange points, while our network has led to the blue points.

Parameter	mean error	median error	95 % of errors lower than
T_{eff} (K)	85	56	247
$\log(g)$	0.11	0.09	0.27
$[Fe/H]$	0.07	0.06	0.16

Table 1. Errors calculated with the validation data set.

Errors on all three fundamental parameters are reported in Tab. 1. The median error on T_{eff} is close to 50 K, and of the order of 0.1 for $\log(g)$. $[Fe/H]$ is affected by a median error of about 0.06. Note that these values are computed from the whole validation data set, but errors can be significantly smaller on sub-domains of our parameter space (e.g. at effective temperatures below 5500 K, see Fig. 3).

This research has made use of the VizieR and SIMBAD tools, CDS, Strasbourg, France.

References

Brewer, J. M., Fischer, D. A., Basu, S., Valenti, J. A., & Piskunov, N. 2015, ApJ, 805, 126
Fabbro, S., Venn, K. A., O'Briain, T., et al. 2018, MNRAS, 475, 2978
Moultaka, J., Ilovaisky, S. A., Prugniel, P., & Soubiran, C. 2004, PASP, 116, 693
Paletou, F. & Zolotukhin, I. 2014, ArXiv e-prints
Petit, P., Louge, T., Théado, S., et al. 2014, PASP, 126, 469
Valenti, J. A. & Fischer, D. A. 2005, ApJS, 159, 141