The Astrophysical Journal Supplement Series, 250:30 (39pp), 2020 October https://doi.org/10.3847/1538-4365/aba8ff

On Neural Architectures for Astronomical Time-series Classification with Application to Variable Stars

Sara Jamal[1] and Joshua S. Bloom[1,2]
[1] Department of Astronomy, University of California, Berkeley, CA 94720-3411, USA
[2] Lawrence Berkeley National Laboratory, 1 Cyclotron Road, MS 50B-4206, Berkeley, CA 94720, USA
Received 2020 March 19; revised 2020 June 9; accepted 2020 July 16; published 2020 October 1

Abstract

Despite the utility of neural networks (NNs) for astronomical time-series classification, the proliferation of learning architectures applied to diverse data sets has thus far hampered a direct intercomparison of different approaches. Here we perform the first comprehensive study of variants of NN-based learning and inference for astronomical time series, aiming to provide the community with an overview on relative performance and, hopefully, a set of best-in-class choices for practical implementations. In both supervised and self-supervised contexts, we study the effects of different time-series-compatible layer choices, namely the dilated temporal convolutional neural network (dTCNs), long-short term memory NNs, gated recurrent units and temporal convolutional NNs (tCNNs). We also study the efficacy and performance of encoder-decoder (i.e., autoencoder) networks compared to direct classification networks, different pathways to include auxiliary (non-time-series) metadata, and different approaches to incorporate multi-passband data (i.e., multiple time series per source). Performance—applied to a sample of 17,604 variable stars (VSs) from the MAssive Compact Halo Objects (MACHO) survey across 10 imbalanced classes—is measured in training convergence time, classification accuracy, reconstruction error, and generated latent variables. We find that networks with recurrent NNs generally outperform dTCNs and, in many scenarios, yield to similar accuracy as tCNNs. In learning time and memory requirements, convolution-based layers perform better. We conclude by discussing the advantages and limitations of deep architectures for VS classification, with a particular eye toward next-generation surveys such as the Legacy Survey of Space and Time, the Roman Space Telescope, and Zwicky Transient Facility.

Unified Astronomy Thesaurus concepts: Variable stars (1761); Periodic variable stars (1213); Light curves (918); Neural networks (1933); Light curve classification (1954)

1. Introduction

Time-domain imaging surveys continue to expand access to the photometric phase space of cadence and depth/volume. Despite many upcoming projects (e.g., the Rubin Observatory Legacy Survey of Space and Time (LSST),[3] Ivezić et al. 2019; Euclid,[4] Laureijs et al. 2011; and the Nancy Grace Roman Space Telescope,[5] Spergel et al. 2015) being optimized for transient (supernovae, microlensing) discovery and characterization, the data from these surveys create unprecedented opportunities to broaden our understanding of stellar variability and stellar evolution as well as expand the use of variable stars (VSs) as probes. Detached eclipsing binaries, for example, provide direct measurements of distance (e.g., Paczyński 1997) and fundamental stellar parameters (Torres et al. 2010), while pulsating VSs such as RR Lyrae, Miras, and Cepheids due to their accurate period–luminosity relations are considered useful tools to trace galactic structures (Kraft & Schmidt 1963; Majaess et al. 2009; Skowron et al. 2019), calibrate the cosmic distance ladder (Freedman et al. 2001; Huang et al. 2018; Riess et al. 2018, 2019), and also act as standard candles to measure distances to their host galaxies (Carretta et al. 2000; Clementini et al. 2003; Alves 2004).

Stellar variability, primarily manifest as changes in brightness and color, arises from various physical mechanisms. Intrinsic variations arise as flares, rotation, pulsations, and/or violent outbursts due to thermonuclear processes occurring in the surface layers or deeper within. Extrinsic factors that may add to the observed variability include eclipses, relativistic Doppler beaming, mutual interaction in binary systems, and/or gravitational lensing.

The classification of VSs is based usually on brightness variations, typically, at visible wavelengths. While far from standard, the General Catalog of Variable Stars (GCVS; Samus' et al. 2017) maintains the taxonomy and nomenclature for VSs that distinguish between subtypes of rotators, pulsators, eruptive variables, cataclysmic variables, and eclipsing binaries in addition to other types such as microlensing sources. In general, stellar variability is not expected to fall into a unique type of dynamical behavior, as objects may display a multitude of physical behaviors, such as rotational modulation superimposed to pulsation in RCB-type stars, rotational modulation interjected by abrupt episodes of deep minima or a steep increase in brightness in BY Dra-type stars, or symbiotic systems with an M-type pulsating Mira star with an accreting white dwarf companion as the R Aqr star. A comprehensive survey of VS classification is provided by Eyer & Mowlavi (2008).

Large data volumes, the primary strength of massive surveys, also present an acute challenge: how do we discover and characterize VSs at scale in streaming, heterogeneous, and noisy time series? Human-free classification, part of a fully automated system to process and analyze survey data, requires the development and deployment of robust and reliable techniques from information-data technology to process large volumes of data and produce tractable and reproducible results.

Traditional machine-learning (ML) approaches for VS classification typically involve "featurization" to summarize

[3] https://www.lsst.org/
[4] https://www.euclid-ec.org/, https://sci.esa.int/web/euclid.
[5] https://roman.gsfc.nasa.gov/

and encode the raw observables into a set of informative descriptors exploited by a classifier to predict labels. Some popular features in the literature include frequency-domain metrics derived from Lomb–Scargle periodograms (Lomb 1976; Scargle 1998), statistical metrics (e.g., standard deviation, quantiles, and skewness), variability indices (e.g., the Stetson indices K and L; Stetson 1996), best-fit model parameters, as well as additional "metadata" information from external catalogs (e.g., colors, redshifts and parallaxes measurements). VS classification using expert-engineered featurization and traditional ML algorithms has been extensively studied and used for decades (Debosscher et al. 2007; Blomme et al. 2011; Dubath et al. 2011; Richards et al. 2011, 2012; Rimoldini et al. 2012; Masci et al. 2014; D'Isanto et al. 2016; Kim & Bailer-Jones 2016). Specialized libraries for features extracting from astronomical light curves have been made available by the community in open-source software packages such as `cesium` (Naul et al. 2016), `FATS` (Nun et al. 2015), `feets` (Cabral et al. 2018), `sncosmo` (Barbary et al. 2016), `gatspy` (VanderPlas & Ivezić 2015; Vanderplas et al. 2016), and `VARTOOLS` (Hartman & Bakos 2016).

Ideally, featurization produces a uniform dimensionality reduced representation of the observations that adequately captures the intrinsic properties of the data needed for classification. However, a known issue in hand-coded feature-based classification lies in the fact that the generated low-dimensional representation may overlook subtleties in higher-order systems and restrict such complexity into a set of low-level descriptors tailored for specific use-case applications. Furthermore, developing domain-specific features can be time-consuming, computationally expensive, highly dependent on expert knowledge, and may show a strong dependency on survey characteristics.

Representation learning (RL) techniques offer an alternative possibility to process raw observables without traditional feature engineering. The benefit of fully automating the classification task using RL lies in the ability to reach a higher level of abstraction and capture complex structures embedded in the data. Distinct approaches in RL to automate feature extraction from astronomical time series have already been introduced in a broad range of studies. Used techniques include unsupervised learning algorithms (Armstrong et al. 2016), dimensionality reduction techniques, data transformations (Johnston et al. 2020), autoencoders (Naul et al. 2018), and dictionary learning (Pieringer et al. 2019).

In recent years, VS classification using deep learning (DL)/neural architectures has been explored in several works that achieved satisfactory classification performance and thus demonstrated the ability of DL systems to learn stellar variability types from light-curve measurements and auxiliary metadata. Among the widely used DL architectures, recurrent neural networks (RNNs) proved to be highly performative for periodic VS classification (Naul et al. 2018; Tsang & Schultz 2019), supernovae (SNe) classification (Charnock & Moss 2017), and online transient event detection (Möller & de Boissière 2019; Muthukrishna et al. 2019a). Convolutional neural networks (CNNs) have also proven comparably preformant, with a better training convergence time and lower memory allocation requirements in comparison to RNNs in various applications such as exoplanet transit detection (Shallue & Vanderburg 2018; Ansdell et al. 2018; Schanche et al. 2019), SNe binary classification (Pasquet et al. 2019b), and Cepheid classification (Dékány et al. 2019). A review on the recent contributions of DL techniques in SNe classification is given by Ishida (2019).

To classify astronomical time series, two main approaches have emerged, either (1) design an automated system to encode the photometric observables into a set of features (semi- or self-supervised learning) that constitute the entry point to traditional algorithms (e.g., support-vector machines, NNs, or tree-based classifiers), or (2) develop a DL architecture to find an optimal mapping between the photometric observables and the labels through a supervised learning scheme.

The objective of this paper is twofold: we first aim to provide an overview of the current ML/DL techniques for VS classification, and second to discuss the applicability of a selected set of NN architectures via a test example and the importance of data representation for classification of stellar variables. This paper also investigates approaches for VS classification using multiband photometric data. The paper is organized as follows. Section 2 first introduces the state-of-the-art ML/DL techniques for VS classification and then proceeds to present our proposed architectures for classification. In Section 3, we present variants of NN architectures and discuss their performances through a test example using public data from the MAssive Compact Halo Objects (MACHO) VS database. The network performances are evaluated in terms of training convergence time, classification accuracy, properties of the generated latent representations, and light-curve reconstruction. Finally, we conclude in Section 4.

2. Deep Learning Architectures for VS Classification

2.1. State of the Art

Classification of VSs can be performed in feature space or in data space. The first approach consists of finding an optimal mapping between the labels, a vector $\boldsymbol{Y}$, and an (encoded) feature set—a matrix $\boldsymbol{X}_{\rm enc}$ derived from the direct observables $\boldsymbol{X}$; the second approach focuses on finding a direct mapping between the observables $\boldsymbol{X}$ and the labels $\boldsymbol{Y}$.

In feature space classification, traditional approaches require hand-coded feature extraction to compute a set of informative descriptors using domain knowledge. For VSs, the most discriminant features of stellar variability depend strongly on the specific class or subclass. For instance, the dominant modes of the pulsation mechanism can be well characterized in the frequency domain, found through Fourier decomposition or periodogram analysis. In eclipsing binaries, the shape, duration, and relative phase of the eclipses in their light curves inform the type of the interaction (contact binaries, detached binaries, or semidetached binaries) and their properties such as the mass and radii ratios. Cataclysmic variables are described through the morphology of their light curves at maximum light, the decline shape, the duration of the burst event, the event occurrence in time distinguishing between recurrent outbursts and final explosions, the quiescent state of the star post-event, and observed spectral features during outburst. For eruptive variables, their light curves and spectra show distinctive variations as sudden brightening or dimming episodes over extended periods of time due to flares and violent processes taking place in the corona or the chromosphere of these stars.

In traditional approaches, feature engineering relies on domain knowledge while feature learning automates the extraction procedure from the data using dimensionality

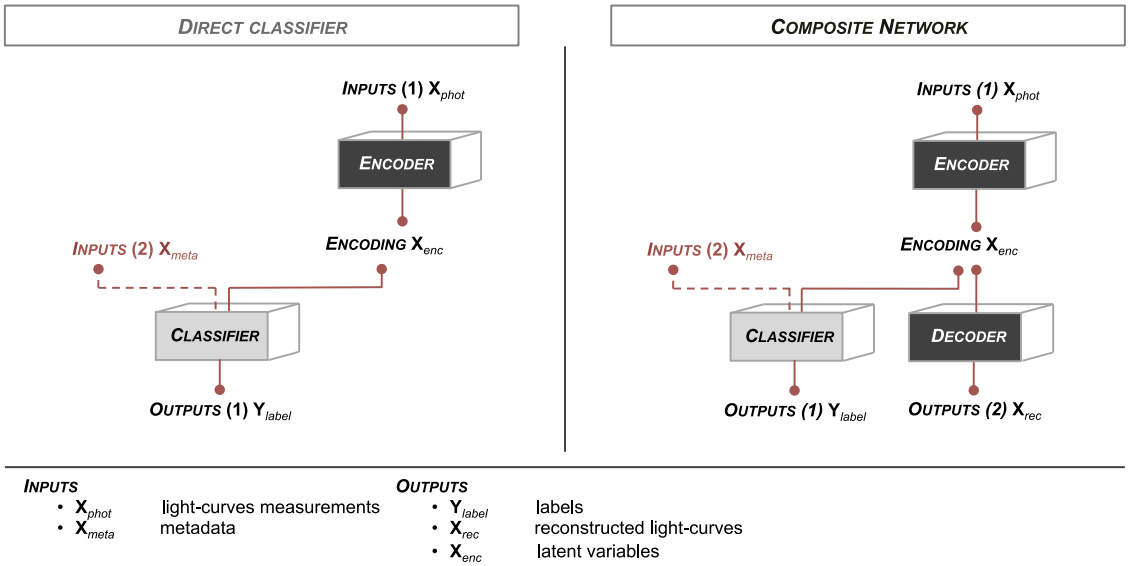

Figure 1. High-level architecture of the direct classifier (left) and composite (right) networks. In direct classifiers, the time-series data ($\boldsymbol{X}_{\rm phot}$) and metadata ($\boldsymbol{X}_{\rm meta}$) are combined through a series of neural layers and concatenations, leading to classification predictions ($\boldsymbol{Y}_{\rm label}$) on which the loss function is optimized. Composite networks use $\boldsymbol{X}_{\rm phot}$ and a bottleneck/decoder to predict a reconstructed light curve ($\boldsymbol{X}_{\rm rec}$). The bottleneck layer along with $\boldsymbol{X}_{\rm enc}$ are also used to predict $\boldsymbol{Y}_{\rm label}$. Both $\boldsymbol{X}_{\rm rec}$ and $\boldsymbol{Y}_{\rm label}$ are used in the loss function of composite networks.

reduction techniques, self-supervised networks (e.g., autoencoders), dictionary learning, or unsupervised algorithms. The extracted features constitute a discretized set of encoded information exploited by feature-based classifiers to predict labels. Among notable references, the work by Armstrong et al. (2016) exploits the unsupervised learning algorithm "Self-Organizing Maps" to encode photometric light curves into a set of features processed, along with additional descriptors, by the tree-based classifier, random forest (RF; Breiman 2001), to predict labels for periodic VSs. The work of Naul et al. (2018) presents a bidirectional RNN autoencoder to discretize the photometric observables into a set of latent variables exploited, along with ancillary metadata, by an RF classifier to predict labels for periodic variables.

For classification in data space, common techniques exploit DL to identify embedded characteristics in the data and find a direct mapping between the input observables and the output labels. Applications using DL techniques for astronomical time-series classification include (1) SNe classification using RNN architectures to process multiband photometric data and auxiliary metadata (e.g., redshift measurements; Charnock & Moss 2017; Möller & de Boissière 2019; Muthukrishna et al. 2019a), (2) online transient event detection using RNN architectures to compute timely class predictions for early-observed light curves in order to forecast potential pre-SN outbursts and prompt follow-up procedures before the event reaches its maximum light (Möller & de Boissière 2019; Muthukrishna et al. 2019a), and (3) exoplanetary transit detection using a composite convolutional network that analyzes the full light curve and the eclipses to discern between planetary transits and stellar eclipsing binaries (Ansdell et al. 2018; Shallue & Vanderburg 2018; Schanche et al. 2019).

More recently, composite architectures have been introduced for astronomical time-series classification in the form of NNs composed of different submodules designed for specific tasks. Notable references include the work by Pasquet et al. (2019b) for SNe classification where the authors propose a DL architecture (PELICAN) with three modules: an autoencoder branch to generate the embeddings at the bottleneck level by optimal reconstruction, a classifier for label predictions, and a contrastive module designed to reduce the discrepancy between the test and train sets. Binary classification of SNe Ia is performed using multiband photometric data and ancillary metadata (redshift measurements of the host galaxies). The work by Tsang & Schultz (2019) proposes a similar approach for periodic VS classification, initially derived from the Deep Autoencoding Gaussian Mixture Model network in Zong et al. (2018). The authors propose a network composed of two modules: (1) an autoencoder, and (2) a classifier module connected to the autoencoder at the bottleneck level.

2.2. Architectures

This section presents selected architectures for VS classification. We distinguish between two types of architectures: direct classifiers and composite networks, as shown in Figure 1. Both architectures are composed of an encoder and classifier module. Composite networks are supplemented with a decoder connected at the bottleneck level. The encoder-decoder combination (i.e., autoencoder) aims to learn a latent representation $\boldsymbol{X}_{\rm enc}$ of the input photometric data $\boldsymbol{X}_{\rm phot}$ by optimal reconstruction, while the classifier maps the encodings to the labels $\boldsymbol{Y}_{\rm label}$. By connecting the autoencoder to the classifier module, the system ideally learns a latent representation (i.e., compressed summary) that is closely correlated to different stellar variability types in the training data. The high-level design in Figure 1 is adaptable: modules can be adjusted to the application and data as needed. In the current work, the decoder module corresponds to a mirror-image of the encoder, which is a common choice for autoencoder architectures. We evaluate the encoder for different types of NNs such as RNNs, CNNs, and their variants, while the classifier module is set to a two-layer multilayer perceptron (MLP) for all

networks to predict labels. A short description of the NNs used in this work is provided in Appendix B.1.

The majority of applications for VS classification using DL exploit auxiliary features that complement the photometric observables such as redshift measurements, detectable frequencies for periodic variables, amplitudes, and colors. The photometric information in the light curves constitutes a fundamental description of the evolutionary state of the star over time but does not contain the entirety of the available information; unless the light curve of a certain class is demonstrably different than other classes, additional metadata can be expected to improve classification accuracy. Typically, classification tasks require an upstream phase of data preparation. However, preprocessing transformations applied to the photometric observables may inadvertently remove discriminating features linked to the stellar variability types, thus altering the quality of the information necessary for classification. For instance, light curves of periodic variables are preprocessed through phase-folding and data normalization. The phase-folding procedure transforms the periodic data into a compact representation in phase of one to two cycles by stacking multiple observations, thus removing the periodicity information over time. On the other hand, data normalization via minmax normalization yields to rescaled magnitude measurements, amplitudes, and errors. As a direct result, similarly shaped light curves with different peak-to-peak amplitudes cannot be distinguished from one another.

In the current work, we investigate the importance of the metadata in two scenarios in which the network classifies the data solely based on (preprocessed) light curves without auxiliary metadata as opposed to supplementing metadata to the system as a secondary input for label predictions.

In VS classification, multiband photometry can be processed either by transforming the photometric passband measurements into a single entity fed to the network or by jointly processing individual encodings from each passband measurement for label predictions. A simplified representation of the aforementioned approaches, identified in this work as the merged and hybrid approaches, is provided in Figure A1. In the merged approach, multi-passband measurements are combined into a unique observable $\boldsymbol{X}_{\rm phot,merged}$ processed by the encoder module to compute the latent representation $\boldsymbol{X}_{\rm enc}$. The preprocessed light curves per band can be combined through distinct representations as the variants presented in Appendix A, whereas the hybrid approach independently encodes the multi-passband measurements into individual features combined at a later time into a compact encoded representation $\boldsymbol{X}_{\rm enc}$. In both scenarios, the classifier module exploits the generated encodings $\boldsymbol{X}_{\rm enc}$, along with metadata $\boldsymbol{X}_{\rm meta}$, for label predictions. Composite networks differ from the direct classifiers by the addition of a decoder module connected at the bottleneck level to the encoder to generate the embeddings by optimal reconstruction.

3. Application

This section presents a set of NN-based architectures for VS classification. We discuss the performance of these networks via a test example in terms of training convergence time, label predictions, reconstruction, and generated latent representations.

Table 1
Selected Data Set from the MACHO VS Database

Class Labels	# in Class
Cepheids FU	1143
Cepheids FO	663
RR Lyrae (type *ab*)	7147
RR Lyrae (type *c*)	1716
RR Lyrae (type *e*)	300
LPV (Wood seq. A)	310
LPV (Wood seq. B)	799
LPV (Wood seq. C)	1100
LPV (Wood seq. D)	756
Eclipsing binaries	3670[a]

Note.
[a] Confirmed binaries in Derekas et al. (2007).

3.1. Data

As an exemplar, for all architectures, we use public photometric data and labels from the MACHO survey (Alcock et al. 1996). The MACHO project carried out long-term photometric monitoring of stars in the Magellanic Clouds and the galactic bulge from 1992 to 1999 in search of rare microlensing events, observable in theory if the dark matter is composed of MACHOs. The large collection of data from the survey has provided, over the years, rare insight into a variety of stellar populations such RR Lyrae, Cepheids, long-period variables (LPVs), and eclipsing binaries (Cook et al. 1995). In MACHO, the LPVs are categorized into four subtypes—namely, the four Wood sequences A, B, C, and D—referring to the parallel sequences identified from the period–luminosity relation of these red variables (Wood et al. 1999). Photometric data in MACHO consists of magnitude measurements in two photometric filters (the MACHO red and blue filters), the associated 1σ error measurements, and the observation epochs expressed in MJD. We exploit the multiband photometric data of the public MACHO VS database[6] to test our selected NN architectures, and we perform further checks on the data. In particular, we retain the confirmed set of eclipsing binaries with corrected periods from Derekas et al. (2007). The full process yields 17,604 periodic variables from the initial count of 21,474 periodic VSs in the MACHO database to test our architectures: 1806 Cepheid variables, 9163 RR Lyrae, 2965 LPVs, and 3670 eclipsing binaries (see Table 1).

The majority of ML/DL open-source software exploits input data in the form of a fixed-size input tensor, and few DL architectures are able to process observables with different lengths, e.g., using generator functions on an iterable list of input data. In our approach, the architecture of the direct classifier can process fixed-size data in a batch mode as well as a list of observables with different lengths using generator functions. Composite networks, however, require fixed-size inputs to comply with the implementation specifications of the decoder module. To meet such requirements, we reduce the data into a fixed-size format. Typically, data reduction can be achieved using padding, rebinning, interpolation (e.g., splines or polynomials), or model fit and prediction. More recently, model prediction using Gaussian Processes (GPs) has been used on astronomical time series (Ambikasaran et al. 2016; Foreman-Mackey et al. 2017; Boone 2019; Pruzhinskaya et al. 2019). To describe the stellar

[6] http://macho.nci.org.au/

variability in the MACHO periodic light curves, a GP model with a quasi-periodic covariance function is fitted to each object using the open-source code by Foreman-Mackey et al. (2019). The selected GP model is a mixture of stochastically driven damped oscillators (SHOs), which are briefly discussed in Appendix B.3. For each object, a GP model is fitted. Using a maximum a posteriori (MAP) estimate, fixed-size light curves are generated by model prediction on a reduced time frame sampled within the range of the observed epochs. The GP predictions' mean and error correspond to the reduced photometric data exploited in the rest of this work. At a later stage, the reduced photometric light curves are normalized and phase-folded to span over two cycles. Phase-folding of periodic light curves allows for a better visualization of the cyclic behavior of the variables. For instance, short-period variables such as RR Lyrae pulsate over timescales ranging from ∼0.3 to 1 day. To fully observe the pulsation profile over a complete cycle, the observation cadence has to exceed the variability frequency to cover a full cycle. In practice, given realistic survey cadences, any one individual cycle will be sparsely observed if at all. By combining the observations through period-folding, however, even high-frequency variables such as RR Lyrae stars can have dense phase coverage. Still, properties of long-period variables that evolve over longer timescales, such as the Mira-type stars with pulsation periods ranging from ∼80 to 1000 days, can be distinguishable in the initial time frame without phase-folding. In the current application, we exploit a data set of periodic VSs with light curves from short-period pulsators, long-period pulsators, and eclipsing binaries and apply the phase-folding procedure to all variables. Observed shortcomings from the preprocessing are discussed in Section 3.3. In the majority of ML applications, normalization is applied to the data to improve the numerical stability and reduce the training time. In this work, the inputs $\boldsymbol{X}_{\rm phot}$ to the networks correspond to preprocessed light curves, obtained after data reduction, phase-folding, and normalization. The metadata $\boldsymbol{X}_{\rm meta}$ consist of the amplitudes, averaged magnitudes, and colors extracted from the raw data in addition to the primary periods.

3.2. Design and Implementation

We experiment with a variety of NN architectures and discuss their absolute and relative performances. Classification using multiband photometry is evaluated using the approaches introduced in Section 2.2, i.e., the merged and hybrid approaches. The merged approach exploits the second representation introduced in Appendix A combining the individual measurements of preprocessed light curves into a 2D tabular format. In the current application, normalization and phase-folding are applied to each band measurement independently, prior to generating the merged representation. Error measurements are used solely as weights in the autoencoder loss function. Incorporating error measurements within the network can be part of further development of such networks. We additionally investigate the use of auxiliary metadata as a secondary input on the classification accuracy for the direct classifier and composite networks via two scenarios, in which the network classifies the data using the information from preprocessed light curves without metadata as opposed to supplementing auxiliary metadata as a secondary input for label predictions. Data entries for the different use-case scenarios are summarized in Table D1. The autoencoder is evaluated for different NNs: RNNs with long-short term memory (LSTM) cells, RNNs with gated recurrent unit (GRU) cells, temporal CNNs (tCNNs), and dilated TCNs (dTCNs).

For VS classification, the objective of NNs is to empirically determine a mapping between the inputs, the photometric observables $\boldsymbol{X}_{\rm phot} = [\boldsymbol{x}_1, \cdots, \boldsymbol{x}_{N_t}]^T$ and auxiliary metadata $\boldsymbol{X}_{\rm meta} = [\boldsymbol{d}_1, \cdots, \boldsymbol{d}_{N_t}]^T$, and the output labels $\boldsymbol{Y}_{\rm label} = [y_1, \cdots, y_{N_t}]^T$, where N_t designates the total number of objects. In the current application, the data vectors $(\boldsymbol{x}_i)_{i:1\rightarrow N_t}$ refer to the reduced photometric measurements—magnitudes, observation epochs, and error measurements—spanning over N_p data points for the ith object, and the metadata $(\boldsymbol{d}_i)_{i:1\rightarrow N_t}$ is composed of N_f features such as the periods, amplitudes, averaged magnitudes, and colors. The elements $(y_i)_{i:1\rightarrow N_t}$ refer to scalar values encoding the labels. For categorical classification, the labels are encoded into codewords transcribing the membership of the object to a variability class $\{j\}_{j:1\rightarrow N_C}$ where N_C is the total number of classes. A standard approach in categorical classification consists of transcribing class memberships into binary codewords $\{0, 1\}$. In online transient event detection applications, a similar methodology is used to map the observables onto the output space. Class predictions are computed on the pixel level, which corresponds to a prediction vector $(\boldsymbol{y}_i)_{i:1\rightarrow N_t}$ allowing one to monitor the evolution over time of the label predictions and forecast potential pre-SN outbursts and trigger follow-up observations within a rapid decision time before the event reaches its maximum light. In contrast, the static-type prediction approach adopted for VS classification consists of associating the static label predictions to fully observed light curves. The approach used in this study aims to map the observables $\{\boldsymbol{X}_{\rm phot}, \boldsymbol{X}_{\rm meta}\}$ onto scalar values $\boldsymbol{Y}_{\rm labels}$. Nonetheless, our architectures can be adapted to perform online predictions on the pixel level by adjusting the input-output format.

Detailed representations of the proposed architectures are shown in Figures D1–D3. After preprocessing (i.e., data reduction, phase-folding, and data normalization of periodic light curves), the networks proceed as follows. In the RNN architectures, the first part of the structure following the input layer is a stack of RNN layers that generates sequence data of fixed length. The direct classifiers and composite networks differ in the last layer of the encoder module in which composite nets are supplemented with a fully connected layer —a dense layer with a linear activation function—to transform the sequence data from the last RNN layer into an encoded representation $\boldsymbol{X}_{\rm enc}$ of a fixed size $N_{\rm enc}$. If auxiliary metadata $\boldsymbol{X}_{\rm meta}$ is supplemented as a secondary input, the classifier exploits the augmented features to predict the labels $\boldsymbol{Y}_{\rm label}$. The classifier module corresponds to an MLP of two dense layers with a rectified linear unit (ReLU) and softmax activation functions. In composite networks, the decoder transforms the encoded variables into reconstructed representations similar to those in Naul et al. (2018). The decoder proceeds by duplicating the embeddings for a number of times that is equivalent to the length of the input data, joining the epoch and passband information, and feeding the augmented embeddings into a stack of RNNs. The last component of the decoder module consists of a fully connected layer that generates the final sequence data of the reconstructed light curves. To

prevent overfitting, regularization is added to the model through the dropout method (Srivastava et al. 2014), which randomly removes units during training.

Following the input layer, the tCNN architecture is composed of a series of convolutional layers with a number of filters and fixed kernel sizes for convolutions. The activation function for each convolutional layer is set to the hyperbolic tangent function, and dropout is used for regularization. In the encoder module, the output of the last convolutional layer is flattened to generate a vector output. The latter corresponds to the final encodings for direct classifiers, while composite networks exploit an additional fully connected layer to generate fixed-size embeddings. Similar to the RNN architectures, the encodings are fed to the classifier module, along with metadata when applicable. In the decoder, the data is processed through a reshaping substructure, stacks of transposed convolutional layers, and a final fully connected layer to generate the outputs. The reshaping substructure emulates the RNN decoder in order to reframe the encodings into a fixed-length format that matches the input data points.

The dTCN architecture closely follows the TCN architecture (Lea et al. 2017) initially derived from the `Wavenet` architecture (Oord et al. 2016). Our design adds dropout functions after the causal convolutions to prevent overfitting. Following the input layer, the first component of the dTCN consists of a causal convolutional layer connected to interconnected stacks of residual blocks with dilated convolutions and gated activation units. Following the `Wavenet` design, an ReLU activation function and two convolutional layers are used after the stacks. The encoder outputs are transformed into a vector format and a fully connected layer is supplemented in composite networks. The generated encodings are then fed into the classifier module and augmented with metadata when applicable. The decoder module is composed of a reshaping substructure, a TCN unit mirrored to the encoder module, and a final fully connected layer.

The NNs are tested with a different set of hyperparameters (see Table D2). The number of parameters per model are provided in Table D3. All models are trained using the `Adam` optimization algorithm (Kingma & Ba 2017) with a learning rate of 5×10^{-4}, a fixed batch size per gradient update in optimization, a dropout fraction, and an early stopping procedure to prevent the networks from overfitting. For all networks, a validation-based early stopping procedure interrupts the training when an optimal solution is found (i.e., convergence) before reaching the maximum number of training epochs. The standard approach monitors the evolution of a scoring metric, typically the validation loss, over a period lag and terminates the training if the error estimated at the current time exceeds the last verified value, implying a large variance in the overfitting regime. An improved search for the hyperparameter space can be performed as an upstream stage of the final classification procedure. However, we choose in this study to empirically evaluate different sets of hyperparameter configurations, discuss the network performances, and identify the best-performing models. In training, the autoencoder (i.e., encoder-decoder) aims to minimize the reconstruction loss and the weighted mean-absolute error (MAE) function, and the classifier branch is set to minimize the categorical cross-entropy loss.

Composite networks are trained to minimize the total loss $L_{\rm tot}$

$$\begin{aligned} \text{direct classifier} \quad & L_{\rm tot} = L_{\rm ec}, \\ \text{composite network} \quad & L_{\rm tot} = w_{\rm ec}\, L_{\rm ec} + w_{\rm ed}\, L_{\rm ed}, \end{aligned} \tag{1}$$

where, $\{w_{\rm ec}, w_{\rm ed}\}$, respectively, refer to the loss weights of the encoder-classifier and the encoder-decoder branches. In the current work, the individual losses $L_{\rm ec}$ and $L_{\rm ed}$ correspond to the following:

$$L_{\rm ec} = \frac{1}{N_t}\sum_{i=1}^{N_t}\left(-\sum_{j=1}^{N_C} y_{{\rm true},i}^{(j)} \log\left(y_{{\rm pred},i}^{(j)}\right)\right), \tag{2}$$

$$L_{\rm ed} = \frac{1}{N_t}\sum_{i=1}^{N_t}\left(\frac{1}{N_p}\sum_{k=1}^{N_p} w_i^{(k)} |x_{{\rm phot},i}^{(k)} - x_{{\rm rec},i}^{(k)}|\right), \tag{3}$$

where, for the ith object, $\boldsymbol{w}_i$ corresponds to the sample weights of the autoencoder branch computed using the inverse of the error measurements σ_i. The losses, averaged across the sample of N_t objects, are computed by, on one hand, averaging the differences between the input photometric light curves $\boldsymbol{X}_{\rm phot}$ and the decoder reconstructions $\boldsymbol{X}_{\rm rec}$ across N_p data points, and, on the other hand, computing the cross-entropy between the true labels $\boldsymbol{Y}_{\rm true}$ and the classifier predictions $\boldsymbol{Y}_{\rm pred}$ over N_C classes. Using the expression for total loss in Equation (1), tested NNs are trained to minimize, at each epoch, the weighted sum of the individual losses with weights $\{w_{\rm ec}, w_{\rm ed}\}$ chosen to be equal to unity, to depict a similar contribution from the individual branches into the total loss. The weighting scheme and cost functions can be revised for different applications and data types. The best-performing models are identified from the minimum loss obtained on a test set, i.e., the subset of data neither used in training nor validation. Our tests used up to two to four cores on a CPU model `Intel Xeon E5-2643v3` on the UC Berkeley Savio Linux cluster. The NN architectures are implemented in the `keras` (https://keras.io) and `Tensorflow` (Abadi et al. 2016a, 2016b) programming frameworks, and the RNN autoencoder branch is partly adapted from the architecture in Naul et al. (2017). To facilitate reuse and reproducibility, our benchmarking codebase is provided in an open-source repository.[7]

3.3. Performance Study

This section presents the results obtained from the public MACHO VS data set. Computations are performed through a classical training-validation-test scheme with 80% of data for training-validation and 20% of unseen objects in the test prediction phase. Performance metrics are computed for all models, with a particular emphasis on the performances reached on the test set. Metrics include the system total loss, the classification accuracy as well as averaged precision, recall, and F1-score (see Appendix C). The total loss corresponds to the classification loss (categorical cross-entropy) evaluated on the encoder-classifier branch, supplemented in composite networks with the weighted MAE evaluated on the encoder-decoder branch. The classification accuracy is obtained through a direct comparison between the true labels $\boldsymbol{Y}_{\rm true}$ and the

[7] https://github.com/sarajamal57/deepnets_vs

network predictions $\boldsymbol{Y}_{\rm pred}$. We also discuss the classification performances within three main types of stellar variability: short-period pulsators including the RR Lyrae and Cepheids, LPVs, and eclipsing binaries. The autoencoder performances are assessed through the quality of the reconstructed light curves $\boldsymbol{X}_{\rm rec}$, and the embeddings $\boldsymbol{X}_{\rm enc}$ are projected onto a 3D representation using a data reduction algorithm. The degree of separation (or lack thereof) in the reduced latent space is discussed.

3.3.1. Training Convergence Time

The training convergence time is reported for all networks in Figure E1. The network type, size, and hyperparameters influence the total time required to reach convergence, with an average training time for RNNs (LSTM and GRU cells) scaling higher in comparison with the convolutional networks (tCNNs and dTCNs) due to higher memory requirements for RNNs that entail a longer time in training. As expected, increasing the network size (i.e., number of parameters) correlates with a longer time in training. For instance, direct classifiers without metadata (c_F) corresponding to the hyperparameter set configuration (*6*), (the largest models for LSTM, GRU, and dTCN) converge in training after approximately 4–5 hr for RNNs, 22 minutes for dTCNs, and ∼6 minutes for tCNNs using one photometric band on CPU. Networks processing multiband data converge at a slower rate due to their larger data entries. After training, the prediction step is extremely fast: 0.5–3 ms per object for tCNNs, 1–10 ms per object for dTCNs, and up to 3–20 ms per object for the RNNs on a CPU.

To track the evolution of the total loss and the accuracy during the training and validation stage, Figures E4–E5 report the performances of the LSTM composite networks d_F and $d_{F,\rm meta}$ using wMAE loss on the autoencoder branch and the categorical cross-entropy on the classifier branch. Over the training epochs, the loss function decreases and converges asymptotically to a constant value whereas the accuracy increases and stabilizes when the system reaches convergence. The system converges to reach at best ∼73% for the best-performing LSTM d_F without metadata. By supplementing the metadata as secondary input, the accuracy increases up to ∼91%. During the validation step, the loss values decrease and moderately exceed the training losses, which reinforces the ability of the networks to generalize the learned mapping from the training to the unseen validation data, as a lack of generalization would correspond to a larger gap between the training and validation losses. Generally, overfitting is detectable from a high variance in the model and a divergence in the validation loss function across training epochs despite the continuing decrease of the training loss. To prevent such limitation, our models are trained using regularization through dropout functions in addition to a validation-based early stopping procedure. Early stopping monitors the validation loss values and interrupts the training before reaching the total number of training epochs if an optimal solution is found (i.e., convergence) or if the system oversees a high increase in the validation losses, indicating a large variance.

We experimented with NN classification using raw (i.e., without normalization) phase-folded light curves. Models converge at a slower rate in an unstable pattern across the training epochs. In what follows, we focus our analysis on the results obtained using preprocessed (i.e., phase-folded and normalized) light curves and associated metadata for MACHO periodic variables stars.

3.3.2. Label Predictions

Total loss and classification accuracy for all trained models are reported in Figures E2–E3. We identify the best-performing models for each architecture type (LSTM, GRU, tCNN, and dTCN) from the minimum loss in the test set, neither used in training nor validation. As previously mentioned, the total loss corresponds to the loss evaluated on the encoder-classifier branch (categorical cross-entropy) supplemented in composite networks to the loss from the encoder-decoder branch (weighted MAE). For RNNs, the losses computed on the validation set are close to the values obtained on the train set across all data sets. However, a larger difference is seen in a few configurations of the tCNN direct classifier processing the multiband data in addition to the configurations of the dTCN direct classifiers. The gap between the validation and the training losses is significant for the dTCNs, which emphasizes the lack of generalization of these types of networks due to a higher complexity (large number of parameters) of the network that is inconsistent with the type of data in hand (1D phase-folded light curves). Complex data would certainly benefit from higher-level network design, as in dTCNs. Conversely, composite networks indicate better stability, due to the addition of the autoencoder contribution to the total loss. We also notice an increase in the classification accuracy for the dTCN composite networks compared to their direct classifier counterparts. In the current application, best classification performances are achieved by RNNs and tCNNs. Table E1 reports the identifiers of the hyperparameter set configurations associated with the best-performing networks. The identifiers correspond to the configuration set identifiers described in Table D2 with varying numbers of layers or stacks {1, 2, 3} and sizes {16, 32}. Using this naming scheme, the identifiers of the best-performing models appear to not fall into a unique hyperparameter set configuration, as the architectures using LSTM layers performed well using one to three layers of different sizes. From Figure E3, the classification results indicate about only a ∼1%–6% dispersion within the individual results per layer type in each of the architectures, which implies that the hyperparameters of the tested networks have a low-to-medium influence on the overall performances. Nonetheless, we notice that most LSTM architectures with one layer ($n_L = 1$) underperform when processing multi-passband data. Deeper networks ($n_L > 1$) appear necessary for processing larger data sets. In the current application, the qualification of the best-performing models is solely based on an empirical search in the network hyperparameter space. Nonetheless, future work could employ an improved search through the hyperparameter space to investigate the effect of hyperparameter selection on the network performances and optimization landscape during training.

Table 2 summarizes the classification accuracy of the best-performing models reached in the test set. For the best-performing models, the overall accuracy ranges between 67% and 79% for models processing the light-curve information without metadata. In particular, without metadata, the dTCNs achieve a classification accuracy of 67%–72% whereas the accuracies of RNNs and tCNNs jointly scale higher with 70%–79% correct predictions. By incorporating the metadata—colors,

Table 2
Classification Accuracy Obtained on the Test Set for the Best-performing Networks (See the Text for a Description) across Three Different Data Sets (*B*-band Only (Top), and Two Variants of the Combination of *R* and *B* Bands (Middle and Bottom))

MACHO – $B_{\rm band}$	ID Net	LSTM	GRU	tCNN	dTCN
	c_F	0.749	0.781	0.732	0.675
	$c_{F,\rm meta}$	**0.916**	0.907	0.887	0.786
	d_F	0.730	0.739	0.701	0.689
	$d_{F,\rm meta}$	**0.905**	0.886	0.900	0.802
MACHO – $RB_{\rm merged}$	ID Net	LSTM	GRU	tCNN	dTCN
	c_F	0.737	0.780	0.722	0.667
	$c_{F,\rm meta}$	0.890	**0.910**	0.815	0.747
	d_F	0.726	0.738	0.691	0.686
	$d_{F,\rm meta}$	0.906	0.883	**0.912**	0.814
MACHO – $RB_{\rm hybrid}$	ID Net	LSTM	GRU	tCNN	dTCN
	c_F	0.776	0.789	0.744	0.696
	$c_{F,\rm meta}$	**0.917**	0.905	0.845	0.768
	d_F	0.748	0.749	0.706	0.726
	$d_{F,\rm meta}$	**0.905**	0.880	0.904	0.818

Note. The best performances among all nets are highlighted in bold.

amplitudes, averaged magnitudes, and periods—in the models $c_{F,\rm meta}$ and $d_{F,\rm meta}$, the accuracy improves by 10%–20% to reach at best: ∼88%–92% for RNNs, ∼82%–92% for tCNNs, and only ∼75%–82% for dTCNs.

The NN architectures learned on single-band and multiband light curves (the blue band *B* versus the multiband *RB*) show comparable classification accuracy. By comparing the individual results in Table 2 for each layer type, the addition of the MACHO red band data injects a moderate effect on the classification accuracy. In particular, the accuracy of the best-performing direct classifiers decreases by 0.1%–3% for RNNs and 1%–7% for tCNNs and dTCNs when using the multiband *RB* data in the *merged* approach in comparison with the *B* band. On the other hand, networks processing the multiband data via the *hybrid* approach mainly achieve a 1%–4% increase in the classification accuracy compared to the *B*-band data. Moreover, the results for the best-performing models do not indicate a significant difference in the individual performances reached by networks processing the multiband photometry *RB* via the *merged* and *hybrid* approaches, as individual results only indicate 0.3%–4% differences, due to the photometric bands (red and blue) in MACHO equally containing informative characteristics on the stellar variability types. This moderate disparity may indicate the need for a different strategy order when combining the data in the *merged* approach, as, in the current application, we combine the preprocessed (i.e., phase-folded and normalized) individual measurements per band. A different strategy can consist of combining the light curves prior to normalization. In general, we would expect that classification using sparsely observed multi-passband photometry would benefit from the availability of several sources of information on stellar variability. In such a case, the advantage of the *hybrid* approach would be more prevalent for systems sequentially processing the multi-passband photometry in an optimized scheme, whereas the *merged* approach may call for higher memory requirements in terms of CPU/GPU usage.

Classification performances are better summarized on a confusion matrix that reports the fraction of predicted labels compared to the true class labels. Optimal results correspond to a diagonal matrix with a fraction of true positives per class (i.e., diagonal elements) close to unity. Figure 2 shows the performance obtained from the test set for the best-performing LSTM direct classifiers c_F and $c_{F,\rm meta}$ that, respectively, reached overall accuracies of 75% and 92%. Overall, the confusion matrices tell a similar story: the main stellar variability groups, highlighted in red, are recovered to a fair accuracy despite the overlap between subtypes and the misclassifications. The network c_F provides class predictions based on the information from preprocessed (i.e., normalized and phase-folded) light curves without metadata. The observed degeneracy is to be expected between classes of objects that share similarly shaped light curves. By supplementing the network with the metadata, the accuracy per class for $c_{F,\rm meta}$ increases, and the number of false positives (i.e., off-diagonal elements in confusion matrices) is significantly diminished. Moreover, the observed porosity between adjacent classes in the confusion matrix appears to remain within the main stellar variability types.

From the confusion matrices, label predictions for the eclipsing binaries in our sample appear to overlap with other variability groups despite the use of the metadata. These misclassifications are possibly due to some degree of (true) label noise that impairs/affects the reliability of the mapping generated from training. In the current tests, class predictions of a few subtypes remain erroneous despite the use of metadata. The second-overtone RR Lyrae pulsators (type *e*) are inaccurately predicted as first-overtone RR Lyrae (type *c*). To avoid confusion within subtypes, a proposed solution would be to introduce a weighting scheme on the feature contributions $\{\boldsymbol{X}_{\rm enc},\ \boldsymbol{X}_{\rm meta}\}$ injecting some prior knowledge regarding the features' importance in VS classification.

We also investigate the accuracy within three main stellar variability groups—short-period pulsators (group 1), eclipsing binaries (group 2), and long-period variables (group 3), and report the classification accuracy achieved in the test set for the best-performing models per group in Table E2. Examining the results of the best-performing RNNs and tCNN, the classification accuracy increases significantly after incorporating the metadata. For short-period pulsators, the accuracy of the best-performing networks reaches 78%–85% without metadata solely based on the distinctive shape of these stars' light curves as the characteristic asymmetry and steep luminosity increase observed in fundamental mode pulsators. With metadata, the accuracy increases up to 92%–94%. The classification of the eclipsing binaries in our sample shows a comparable improvement, reaching 91%–92% correct predictions at best when using metadata. Similarly, the LPV sample benefits from the use of metadata as a secondary input, as the best-performing networks

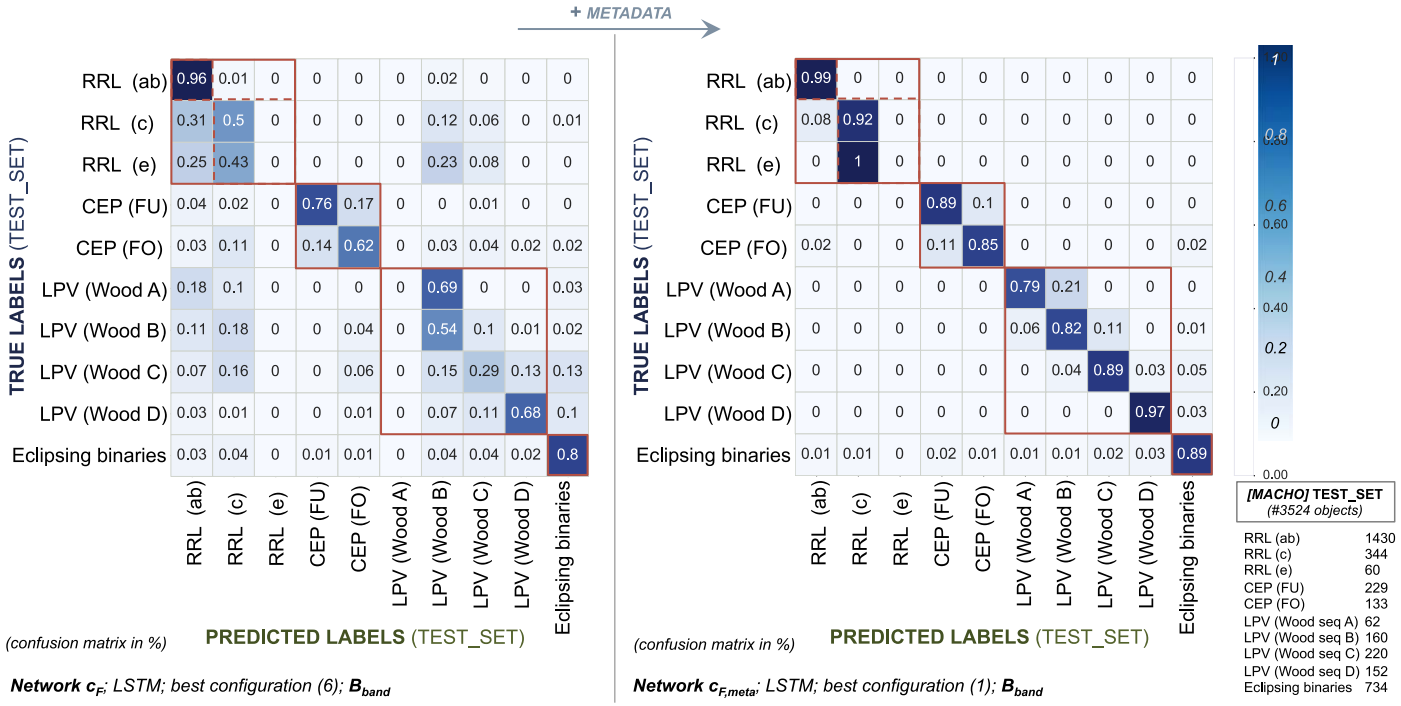

Figure 2. Confusion matrices obtained from the test set predictions for the best-performing LSTM direct classifiers c_F and $c_{F,\mathrm{meta}}$ on the B band. The values in each box correspond to the number of predictions vs. the initial count of true labels (indicated on the right). The main stellar variability groups are highlighted in red, and comprise the RR Lyrae, Cepheids, LPVs, and eclipsing binaries.

achieve 65%–88% of true positives at best from an initial 30%–54% without metadata.

To further investigate the classification performances, we compute additional metrics (see Table E3). The recall (i.e., the true positives rate or sensitivity) characterizes the ability of the network to properly retrieve the true labels, and the precision depicts the level of agreement between the predictions per class and the true labels. The F1-score corresponds to a combination of both metrics. We report in Table E4 the averaged metrics for the best-performing LSTM direct classifiers c_F and $c_{F,\mathrm{meta}}$ that, respectively, showcased classification accuracies of 75% and 92%. The averaged precision, recall, and F1-score are boosted, respectively, to 77%, 80%, and 78% from an initial ∼51% rate. Individual metrics per class highlight an increased improvement. However, the inability of the network to predict a few subtypes such as the second-overtone RR Lyrae pulsators affects the recall and precision averaged across all observations in the test set. Table E3 reports the averaged metrics for the best-performing models for the GRUs, tCNNs, and dTCNs. Similar conclusions can be reached regarding the performances of the RNNs and tCNNs outperforming the dTCNs in the current tests.

To characterize the need for metadata and photometric observables for classification, we performed a supplementary classification test using only the metadata as inputs to the classifier module. The results are reported in Table E5. Using the metadata (i.e., the amplitudes, averaged magnitudes, periods, and colors), the network achieves an 83% accuracy (i.e., fraction of true positives). However, the classification metrics per class indicate a high number of false predictions in addition to the inability to predict some subtypes, which corresponds to a lower precision, recall, and F1-score compared, for instance, to the performances of the best-performing RNN direct classifier and composite network in Table E3. Overall, combining the information encoded in the light curves and the metadata allows for a better mapping characterizing the stellar variability types.

3.3.3. Reconstructed Light Curves

In this section, the autoencoder performance is assessed through the quality of reconstructed light curves $\boldsymbol{X}_{\mathrm{rec}}$. An ideal reconstruction would preserve the embedded (denoised) structure of the input data $\boldsymbol{X}_{\mathrm{phot}}$. To visually assess the reconstruction quality, we select a sample of objects for display (see Table 3) and show the reconstructed light curves for the best-performing composite network $d_{F,\mathrm{meta}}$ in Figures E6–E7. A subset of the reconstructed light curves for the best-performing LSTM composite network is also shown in Figure 3.

Overall, the reconstruction results indicate a smoothing effect on the magnitude measurements as well as a correlation between the decoder performance and the input data quality. In particular, low signal-to-noise levels in the data limits the ability of the network to recover real structures, and the resulting $\boldsymbol{X}_{\mathrm{rec}}$ appears exaggeratedly smoothed out with no distinct features (such as the characteristic pulsation profile or peaks at maximum light). Furthermore, some pulsating variables can exhibit irregularity or low-frequency modulations that may evolve on timescales longer than the observed time range, noticeably seen for long-secondary periodic (LSP) stars in the Wood sequence D (Wood et al. 2004). In such cases, the folded light curves using the primary period appear as a mismatched superimposition of multiple cycles, as seen in the irregular LPV displayed in Figure 3. To prevent such limitations, detection of multi-periodicity, irregularity, and low-frequency modulations should be considered to potentially isolate those objects that may require a different preprocessing strategy and classification approach.

Table 3
Selected Subset of Objects from the MACHO VS Database for Display

	Object ID[a]	Variability Type	α (rad)	δ (rad)	Period (days)	$\langle R \rangle$ (mag)	$\langle B \rangle$ (mag)	$\langle K_V \rangle$ (mag)	$\langle K_R \rangle$ (mag)	$\langle K_V \rangle - \langle K_R \rangle$ (mag)	Amplitude (in R)	Amplitude (in B)	SSR[b]
n1	82.9138.798	RR Lyrae (type ab)	1.46723	−1.20004	0.539	−4.342	−4.198	20.096	19.692	0.403	0.729	1.053	0.416
n2	1.3449.1187	RR Lyrae (type c)	1.31759	−1.20170	0.337	−4.829	−4.906	19.427	19.244	0.183	0.258	0.399	0.652
n3	1.4052.2961	RR Lyrae (type e)	1.33093	−1.20398	0.264	−4.849	−4.848	19.471	19.21	0.261	0.226	0.396	1.005
n4	1.3441.45	CEP (FU)	1.31567	−1.21106	3.362	−8.420	−8.226	16.059	15.605	0.454	0.505	0.715	0.063
n5	81.9241.38	CEP (FO)	1.47038	−1.22092	1.973	−7.797	−7.420	16.832	16.195	0.637	0.217	0.320	0.208
n6	1.4046.1610	LPV (Wood seq. A)	1.33270	−1.21091	45.150	−9.351	−8.220	15.896	14.505	1.391	0.085	0.086	0.928
n7	5.4407.15	LPV (Wood seq. B)	1.34311	−1.21385	131.004	−9.397	−7.895	16.154	14.392	1.762	0.169	0.251	0.959
n8	82.9134.20	LPV (Wood seq. C)	1.46758	−1.20433	267.152	−7.578	−5.348	18.570	16.080	2.490	1.462	2.424	0.271
n9	1.3689.30	LPV (Wood seq. D)	1.32271	−1.20431	826.788	−8.949	−7.259	16.756	14.806	1.949	1.086	1.346	0.467
n10	1.3442.233	Eclipsing binary	1.31643	−1.21027	1.640	−6.332	−6.725	17.665	17.798	−0.132	0.930	1.090	0.208

Notes.
[a] The standard three-integer identifier in the MACHO photometry database (field.tile.sequence).
[b] Model-fit residuals from `Supersmoother` applied to the reduced photometric B-band data.

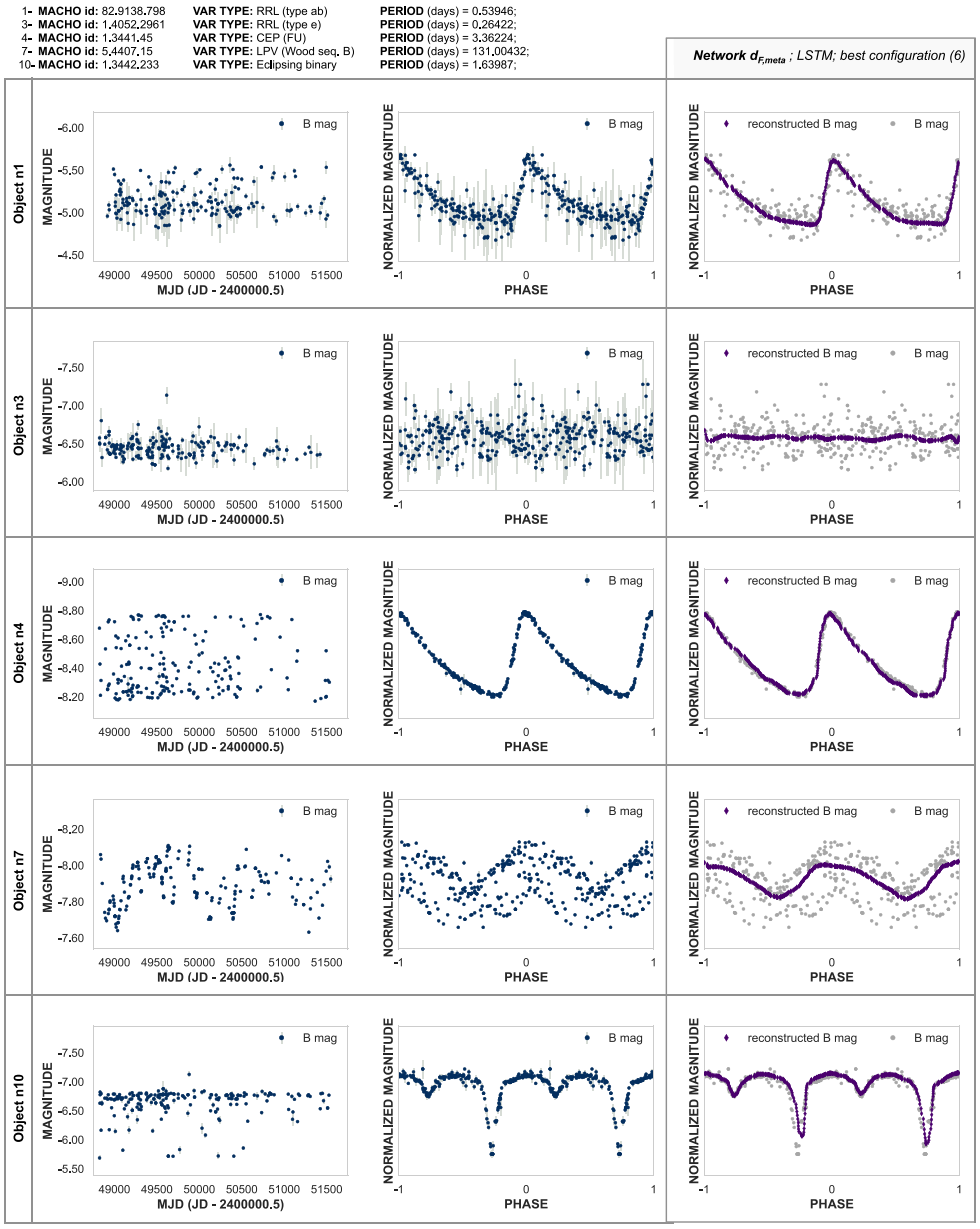

Figure 3. Reconstructed light curves from the test set for the best-performing LSTM composite network $d_{F,\mathrm{meta}}$.

To assess the global performances reached on the test set, we evaluate the reconstruction error as a function of a data-quality indicator (SSR) corresponding to the model-fit residuals computed from the `SuperSmoother` algorithm in Friedman (1984). The `SuperSmoother` performs a component-wise linear smoothing of the time-series data using adaptive bandwidths. The residuals are obtained from the averaged differences between the time series and the regression fits. In the current application, we use the SSR as a direct indicator of the data quality in order to discuss the decoder reconstructions in the tested NNs. We expect the light-curve measurements with low signal-to-noise, irregular variations, or extended low-frequency modulations in time to be associated with a high SSR. The distribution of the reconstruction error in Figure 4 suggests a distinct trend: the system is able to moderately recover the morphology of objects associated with low SSR (e.g., the Cepheids n4 and n5 and the Mira star n8) as opposed to noisier and irregular light-curve profiles (e.g., objects n3, n6, and n7, which are RR Lyrae of type *e* and LPVs from the Wood sequences A and B).

A comparable analysis on the performances of the best-performing GRU, tCNN, and dTCN composite models $d_{F,\mathrm{meta}}$ reaches a similar conclusion on the distribution of the reconstruction error (see Figure E8). From the reconstructed profiles of the selected objects (see Figures E6–E7), networks achieve overall comparable performances despite few noticeable differences, such as the reconstruction of the shockwave propagating before the maximum light in the fundamental model RR Lyrae (object n1) that is recovered by the convolutional networks but heavily smoothed out in the LSTM and GRUs. The RNNs also appear to overly smooth out the modulations of the LPV star (object n9, LPV Wood sequence D), while convolutional nets partially restore discontinuous plateaus. Furthermore, the composite LSTM model appears to preserve the shape of the primary and secondary eclipses in the detached eclipsing binary (object n10), while convolutional nets (dTCNs and tCNNs) partially recover the depth of the eclipses and the GRU disproportionately smoothing out these features. Based on the reconstructed profiles of selected objects, the moderate smoothing of small-scale features by the

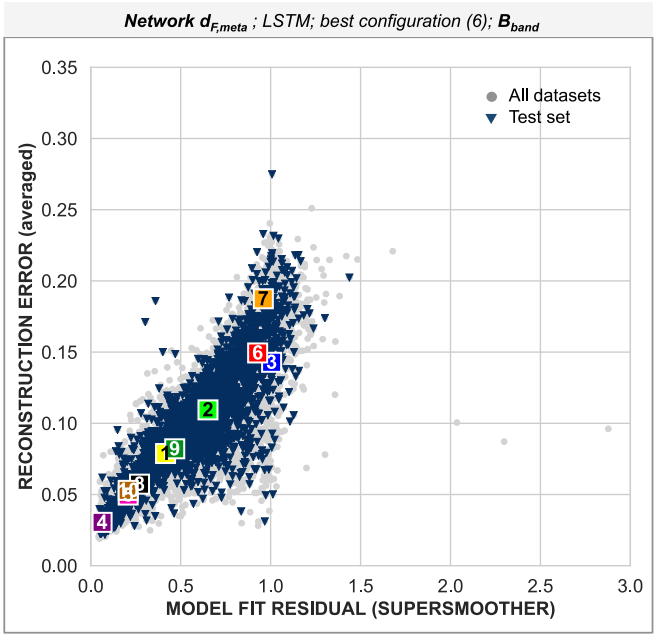

Figure 4. Reconstruction error (MAE) as a function of the model-fit residuals from the `SuperSmoother` algorithm (Friedman 1984) for the best-performing LSTM $d_{F,\mathrm{meta}}$ on the B band. The highlighted numbers (1–10) refer to the subset of selected objects from the test set used to showcase the reconstruction quality of the autoencoder branch.

RNNs can be interpreted as these models focusing on an overall data structure propagated through the RNN cells. The signal-to-noise of these features plays a role as well in the reconstruction quality. Convolutional NN variants, given the choice of kernel sizes for convolutions, exhibit a similar behavior through a localized smoothing in the reconstructed profiles.

From the reconstruction error distributions, the reconstruction performances of the best-performing LSTMs, GRUs, tCNNs, and dTCNs show a comparable reconstruction ability. In our current designs, the decoder outputs reconstructed magnitudes. To further characterize the ability of the decoder modules to unfold the generated embeddings into reconstructed profiles close to the input data, the development of novel designs to output reconstructed profiles along with the associated prediction errors is left for future study.

3.3.4. Latent Space Exploration

Using dimensionality reduction algorithms, encoded features of the train set can be projected onto a reduced (2D or 3D) representation. The results for the best-performing LSTM composite network $d_{F,\mathrm{meta}}$ are shown in Figure 5. For better visualization, projections are separated for the three main variability groups in Figures E9–E11. We limit the analysis of the latent representation to the training data set as it corresponds to the learned partitioning.

The encoded features generated from the best-performing composite network are projected onto a 3D representation using the Uniform Manifold Approximation and Projection (UMAP) algorithm (McInnes et al. 2018) described in Appendix B.2. The degree of separation of the clusters in the reduced representation space characterizes the type of information fed to the classifier network. Without metadata, solely based on the encoded morphology of the light curves, the projection outlines (see left panel of Figure 5) a large fraction of RR Lyrae, Cepheids, eclipsing binaries, and LPVs isolated to some extend in the projected space. The level of separability of these clusters is limited by the overlap between classes of objects sharing a similar shape of (preprocessed) light curves, as noticed for the majority of LPVs, overtone pulsating RR Lyrae, Cepheids, and a few eclipsing binaries in our sample. In the detailed representations (see Figures E9–E11), the encoded features of eclipsing binaries are clustered into composition of a compact aggregate, a dispersed set, and outliers, while short-period pulsators cluster into a compact aggregate of fundamental mode pulsators and an overlapping blend of overtone pulsators, given their similar light-curve profiles. In the latent space, the LPV sample clusters into a compact aggregate without a clear delineation between the different subtypes; this suggests a lack of discriminating features in $\boldsymbol{X}_{\mathrm{enc}}$ that would be necessary to distinguish the different subtypes.

When metadata is supplemented, the projection of the augmented features set $\{\boldsymbol{X}_{\mathrm{enc}}, \boldsymbol{X}_{\mathrm{meta}}\}$ shows a better separability in the reduced latent space (see Figure 5 on the right); this separability also coincides with the improvement in the classification accuracy for the networks utilizing metadata to complement the encoded photometry. In particular, the detailed representations for the main variability types highlight well-separated clusters for the short-period pulsators, while the sample of eclipsing binaries clusters into a composition of a compact aggregate, a filamentary structure, and dispersed outliers. The diversity in substructures in the eclipsing binaries sample can be explained by the different categories of binaries merged in the MACHO database (e.g., contact, detached, and semidetached stellar binaries) in addition to possible label contamination. Similarly, the LPV sample is well separated from other variability groups and is projected onto a filamentary structure with an enhanced separability between the different subtypes. The best-performing GRU, tCNN, and dTCN composite networks give comparable projection results.

To investigate the properties of the embeddings obtained from composite networks versus direct classifiers, a projection of the generated encodings obtained by the best-performing LSTM direct classifier is reported in Figure E12. Compared to the composite network, the direct classifier network generates an embedding layer X_{enc} by propagating the information between an encoder and a classifier module. In the latent representation, distinct clusters are noticeable along with the expected overlap of objects with a similar light-curve shape. From the 3D representation, the main difference between the embeddings generated from the LSTM direct classifier (see Figure E12) and the LSTM composite network (see Figure 5) lies in the clusters' (intraclass) dispersion. The encoder-decoder combination in the composite network appears to narrow the clusters in the latent representation; this effect would be useful for anomaly detection to locate potential outliers at the outskirts of identified compact aggregates or the intersection of adjacent clusters.

To summarize, composite NN architectures are able to encode the photometric observables into substructures associated with the stellar variability classes. Without metadata, the encodings generated from the photometric data cluster into distinct aggregates despite the overlap between classes of objects sharing a similar morphology of preprocessed light curves. By supplementing the metadata as a secondary input to complement the encoded photometry, the level of separation in the latent space is enhanced, which aligns with the overall increase in classification accuracy. An examination of the nature of the overlap regions of the latent space, as well as the

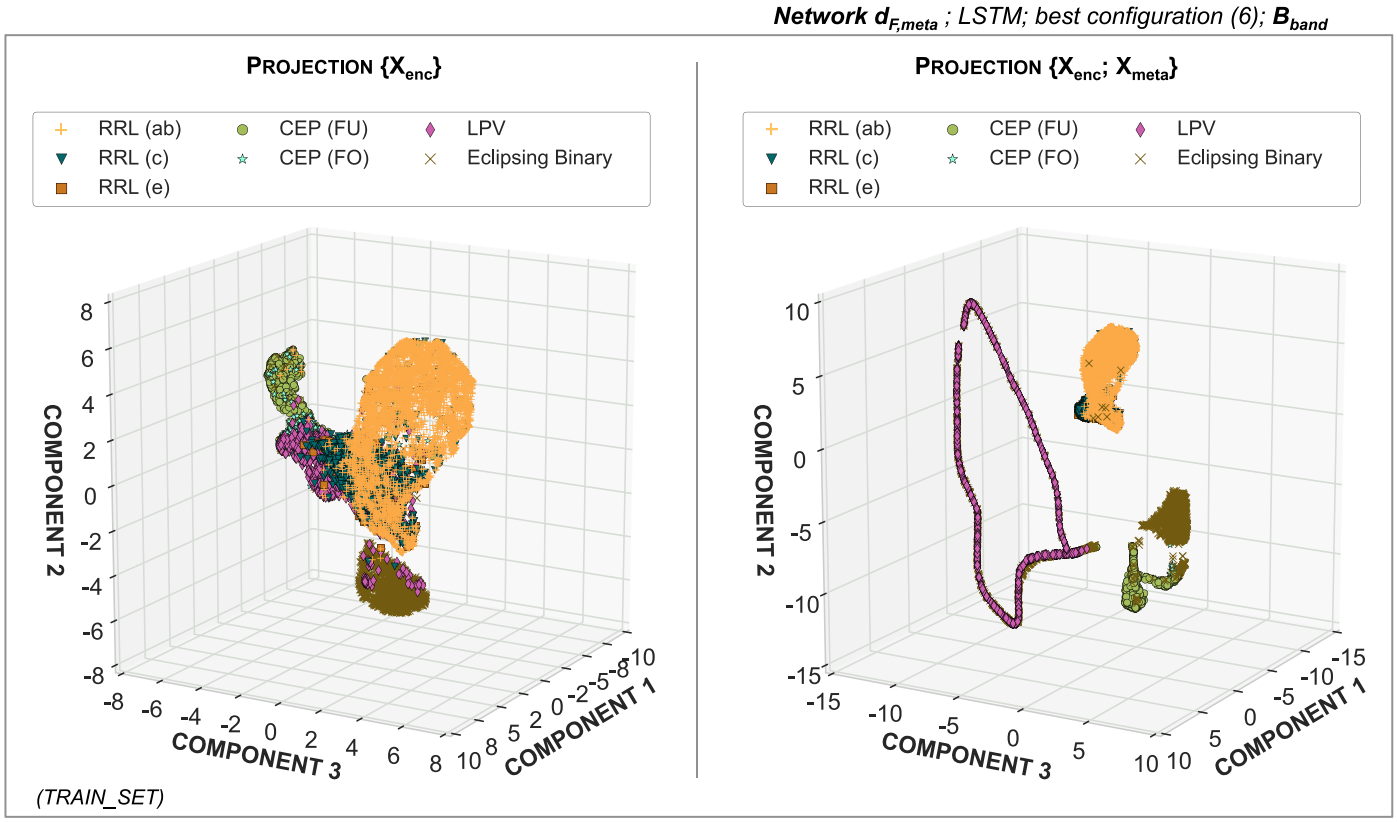

Figure 5. Three-dimensional representation of encoded features from the best-performing LSTM composite network $d_{F,\mathrm{meta}}$ on the B band. Generated encodings are projected into a reduced 3D representation using the UMAP algorithm.

properties of the different aggregates, is left for future study. We expect that similar studies using larger VS data sets will help test the potential universality of the latent space, and also reveal potential outliers that could constitute new subclasses.

4. Conclusions

In this work, we explored the use of NN architectures for VS classification through various use-case scenarios. These architectures allowed us to generate higher-abstraction encodings of the photometric data without the need for hand-coded feature engineering.

Two types of architectures, identified as direct classifiers and composite networks, were tested. Both networks are composed of an encoder module to transform the data into a reduced representation and a classifier to predict labels. Composite networks include a decoder module to define an encoded representation of the input data by optimal reconstruction. In our analysis, VS classification using multi-passband photometric data was performed by two approaches, either by encoding a merged representation of all passband measurements (*merged* approach) or jointly processing individual encodings (*hybrid* approach). Sparsely observed multi-passband photometry would benefit from adopting the latter approach.

In this work, we also experimented with a variety of NN architectures and investigated the effect of ancillary metadata on classification performance. Through an empirical search on different hyperparameter set configurations, the best-performing models were identified. Models exploiting hyperparameter tuning through optimized ML approaches or Bayesian optimization are left for future work. In our work, we found that systems solely exploiting the time-series data were able to reach a ∼70% accuracy for the best-performing models. By supplementing the metadata as a secondary input, a net increase in the classification accuracy is observed across all network types, reaching at best a ∼91% accuracy for the best-performing LSTMs and temporal CNN models. Misclassifications for the best-performing networks are primarily restricted to the main stellar variability types, which provides a strong incentive for a multistage architecture for label predictions, to first predict the main stellar variability type, followed by subtype prediction. On a computational level, the training convergence time for RNN models was found to be longer, due in part to larger memory allocation costs.

For composite networks, the reconstruction quality of the decoder module appears to be highly contingent on the input data properties (i.e., the signal-to-noise level and smoothness of the light-curve profiles). VSs exhibiting multi-periodicity, irregularity, or modulations over time appear in their phase-folded representation as a mismatch of superimposed cycles that a network is unable to learn and overly smooths out in reconstruction.

The exploration of the learned encodings indicated a clear clustering linked to the stellar variability types. Without metadata, clusters of variables appear isolated despite the overlap noticed for objects sharing similar light-curve profiles. Supplementing the encoded information with the metadata predictably lessens such degeneracy and enhances the separation between the classes; this, in turn, accounts for the increase in the fraction of correct predictions. We would expect the latent representations to highlight interesting properties in the data and pinpoint potential outliers, unknown stellar variability types, or new subtypes within known variability classes.

To conclude, various NN architectures are able to capture low-dimensional data representations and achieve excellent classification accuracy without the need for hand-coded featurization. The best-performing networks in our tests are primarily LSTM- and tCNN-based models, with the latter benefiting from smaller training convergence time and smaller memory footprints.

As part of future work, the development of a baseline for an automated system able to learn a wider range of stellar variability traits can be explored. The need for general architectures is strongly motivated by the fact that massive surveys are set to produce large data sets with a blend of different types of stellar variables such as aperiodic VSs (e.g., cataclysmic stars and microlensing events) as well as periodic and quasi-periodic variables (e.g., pulsators, rotators, and eclipsing binaries). The automated classification for periodic VSs presented here exploits phase-folded representations as well as information from the frequency domain, while classification of quasi-periodic variables requires the use of a combination of multiple data representations—phase-folded light curves, periodograms, $O-C$ diagrams, and time series—to produce reliable class predictions. To meet the need for a general framework, one proposed design would consist of a multistage architecture with different components specialized to distinguish distinct stellar variability traits, to first discriminate between the three categories of periodic, quasi-periodic, and aperiodic VSs, then follow with classification into the stellar variability types and subtypes.

Despite our analysis being focused on applications for periodic VS classification, all arguments presented in the scope of this work extend to other types of astronomical time series. NNs can be built with a comparable architecture for supernovae classification and transient detection, with adaptations of the input data representation, the preprocessing strategy, and the necessary metadata (e.g., redshift measurements and spectral features). Similarly, our approaches in processing multi-passband photometric data for classification can be generalized to other variable objects. On the network design, the complexity of the networks should conform to the type of data. In particular, higher-level multidimensional data would require deeper and complex architectures compared to 1D information such as phase-folded light curves. Using the presented methodology on different data sets, we would expect, on one hand, an increase in the classification accuracy when supplementing the metadata, and on the other hand, a significant improvement in classification when combining sparse observations across multiple photometric bands. For sparsely observed light curves, the networks processing multiband data would very likely exceed the classification results obtained with a single photometric band. We expect the landscape of the latent space representation to differ from the current results. Further analysis of latent representations obtained from a larger scope of variability types is left for future study.

This research made use of the Savio computational cluster resource provided by the Berkeley Research Computing program at the University of California, Berkeley (supported by the UC Berkeley Chancellor, Vice Chancellor for Research, and Chief Information Officer). S.J. and J.S.B. were partially supported by a Gordon and Betty Moore Foundation Data-Driven Discovery grant. The authors would like to thank the referee for helpful comments and multiple suggestions that significantly improved this manuscript. We would also like to acknowledge and thank J. Martínez-Palomera for his careful readings and helpful comments. This paper utilizes public domain data obtained by the MACHO Project, jointly funded by the US Department of Energy through the University of California, Lawrence Livermore National Laboratory under contract No. W-7405-Eng-48, by the National Science Foundation through the Center for Particle Astrophysics of the University of California under cooperative agreement AST-8809616, and by the Mount Stromlo and Siding Spring Observatory, part of the Australian National University.

Software: `NumPy` (Oliphant 2006; van der Walt et al. 2011), `scikit-learn` (Pedregosa et al. 2011), `TensorFlow` (Abadi et al. 2016a, 2016b), `keras` (https://keras.io), and `exoplanet` package (Foreman-Mackey et al. 2019) and its dependencies for GP model prediction: `celerite` (Foreman-Mackey et al. 2017; Foreman-Mackey 2018), `pymc3` (Salvatier et al. 2016), and `theano` (Theano Development Team et al. 2016).

Appendix A
Classification Using Multiband Photometric Data

This section details our approaches for classification with NNs using multiband photometric data. We represent, in Figure A1, the two approaches identified as *merged* and *hybrid* approaches. In the *merged* approach, the multi-passband photometric data can be combined into a 2D tabular data of dimension $(N^{*}_{p,\mathrm{merged}} \times (2m+1))$, in which the m photometric band magnitude and associated error measurements are provided along with the observation epochs.

The ith observation in $\boldsymbol{X}_{\mathrm{phot,merged}}$ corresponds to the matrix representation:

$$\boldsymbol{x}_i = \begin{pmatrix} t_i^{(1)}, [\mathrm{mag}_i^{(1)}, \sigma_i^{(1)}]_{\mathrm{band}\ 1}, & \cdots & [\mathrm{mag}_i^{(1)}, \sigma_i^{(1)}]_{\mathrm{band}\ m}, \\ \vdots & \vdots & \vdots \\ t_i^{(P^*)}, [\mathrm{mag}_i^{(P^*)}, \sigma_i^{(P^*)}]_{\mathrm{band}\ 1} & \cdots & [\mathrm{mag}_i^{(P^*)}, \sigma_i^{(P^*)}]_{\mathrm{band}\ m} \end{pmatrix}, \tag{A1}$$

where m refers to the number of photometric bands and P^* designates the total count of data points $N^{*}_{p,\mathrm{merged}}$ for the ith observation. In this representation, the sparsity level of the matrix depends on the observation times across the different photometric bands.

Alternatively, multi-passband data can be combined into a 2D tabular data of dimension $(N_{p,\mathrm{merged}} \times 4)$, in which an auxiliary vector encoding the photometry band is provided along with the observation epochs and the m band magnitude and associated error measurements. The encoding vector associates each photometric band type with a dictionary item (numerical or qualitative variables).

$$\boldsymbol{x}_i = \begin{pmatrix} \mathrm{dict}_{\mathrm{band}\ 1}, \boxed{[t_i, \mathrm{mag}_i, \boldsymbol{\sigma}_i]}_{\mathrm{band}\ 1} \\ \vdots \\ \mathrm{dict}_{\mathrm{band}\ m}, \boxed{[t_i, \mathrm{mag}_i, \boldsymbol{\sigma}_i]}_{\mathrm{band}\ m} \end{pmatrix}, \tag{A2}$$

where the enclosed measurements per band $\{b\}_{b:1\rightarrow m}$ correspond to a matrix of $(P_b \times 4)$ dimension as follows:

$$\boxed{[t_i, \mathrm{mag}_i, \boldsymbol{\sigma}_i]}_{\mathrm{band}\ b} = \begin{pmatrix} [t_i^{(1)}, \mathrm{mag}_i^{(1)}, \sigma_i^{(1)}]_{\mathrm{band}\ b} \\ \vdots \\ [t_i^{(P_b)}, \mathrm{mag}_i^{(P_b)}, \sigma_i^{(P_b)}]_{\mathrm{band}\ b} \end{pmatrix}, \tag{A3}$$

where $\{P_b\}_{b:1\rightarrow m}$ refers to the data point count per band.

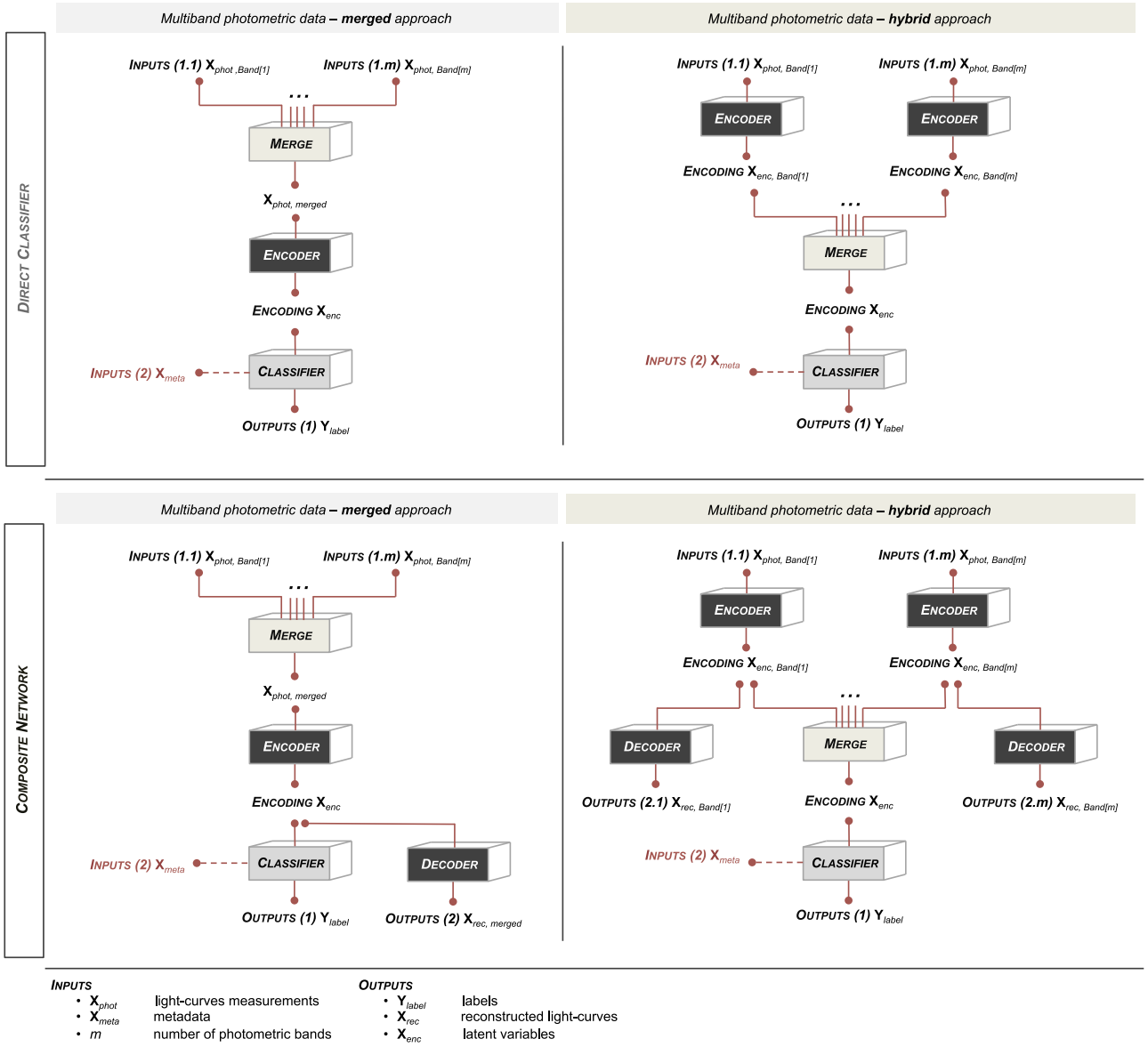

Figure A1. High-level architecture of the direct classifier (top) and composite (bottom) networks across two variants of the combination of multi-passband photometric data, identified as *merged* and *composite* approaches (left and right). In the *merged* approach, multi-passband measurements are merged into a unique observable $\boldsymbol{X}_{\rm phot,merged}$ processed by the encoder module to compute the latent representation $\boldsymbol{X}_{\rm enc}$, whereas the *hybrid* approach independently encodes the multi-passband measurements into individual features merged at a later time into a compact encoded representation $\boldsymbol{X}_{\rm enc}$. In both scenarios, the classifier module exploits the generated encodings $\boldsymbol{X}_{\rm enc}$, along with metadata $\boldsymbol{X}_{\rm meta}$, for label predictions. Composite networks differ from the direct classifiers by the addition of a decoder module connected at the bottleneck level to the encoder to generate the embeddings by optimal reconstruction.

Appendix B
Description of NNs, UMAP, and Gaussian Process Modeling

B.1. RNNs, CNNs, and TCNs

RNN refers to an NN architecture composed of interconnected nodes through a directed graph (cyclic or acyclic) along a temporal sequence. In the standard architecture, the fully connected RNN layer is constructed such that each node is interlinked to the adjacent units. Each node in the standard architecture corresponds to a neural unit with an activation function and a weight. The LSTM (Hochreiter & Schmidhuber 1997) and the GRU (Cho et al. 2014) are variants of the RNN with a higher-level node structure composed with multiple subunits acting as internal regulators to the propagated information within the network. Both the LSTM and GRU cells exploit a forget gate and an input gate, and the LSTM utilizes an additional output gate. RNN applications to astronomical time series include SNe classification (Charnock & Moss 2017), VS classification using autoencoders (Naul et al. 2018), gravitational-wave signal denoising (Shen et al. 2019), periodic VS classification (Tsang & Schultz 2019), and online transient event detection (Möller & de Boissière 2019; Muthukrishna et al. 2019a).

CNN refers to an NN architecture with convolutional layers that applies a series of convolutions through overlapping windows, allowing one to capture spatial correlations in the data. The standard CNN architecture utilizes pooling layers after convolutions to downsize the data in addition to fully connected layers. The performances of convolutional-based

NNs have been demonstrated in a broad range of astronomical data applications such as galaxy classification (Dieleman et al. 2015; Aniyan & Thorat 2017; Kim & Brunner 2017; Domínguez Sánchez et al. 2018), VS classification using asteroseismology (Hon et al. 2017), supernovae classification (Cabrera-Vives et al. 2016; Brunel et al. 2019; Pasquet et al. 2019b), photometric redshift estimation (Hoyle 2016; D'Isanto & Polsterer 2018; Pasquet et al. 2019a), cosmological parameter inference (Ntampaka et al. 2020), parameter estimation from 21 cm tomography (Gillet et al. 2019), strong lensing detection (Lanusse et al. 2018; Jacobs et al. 2019), gravitational-wave signal detection (Gebhard et al. 2017, 2019; Gabbard et al. 2018; George & Huerta 2018a, 2018b; Fan et al. 2019), generator models for weak lensing convergence maps (Mustafa et al. 2019), cosmic-ray modeling (Erdmann et al. 2018), detection of damped Lyα systems in quasar spectra (Parks et al. 2018), fast radio burst classification (Connor & Leeuwen 2018), and classification of supernovae spectra (Muthukrishna et al. 2019b).

TCN refers to an NN architecture in Lea et al. (2017), initially derived from the `Wavenet` architecture (Oord et al. 2016). The network is a composition of a series of dilated convolutions and residual blocks used to expand the filters' receptive fields and reduce the training convergence time. A detailed description of deep learning techniques is available in specialized computer science publications, such as the overview of DL techniques by Schmidhuber (2015).

B.2. UMAP

The UMAP algorithm is a nonlinear dimensionality reduction algorithm introduced by McInnes et al. (2018). Assuming a uniform distribution of the data on a locally connected Riemannian manifold, the algorithm computes a low-dimensional representation by optimizing a fuzzy set cross-entropy between the simplicial set representations of the data and the target embeddings. The UMAP has gained interest and has been used recently for astronomical data applications such as the SDSS DR15 spectroscopic data classification in Clarke et al. (2020) and anomaly detection in SDSS galaxy samples in Reis et al. (2019).

B.3. Model Prediction Using Gaussian Processes

Part of preprocessing, data reduction is performed using a GP model to generate fixed-length representations of each source. Foreman-Mackey et al. (2017) provided GP kernels suitable for astronomical time series, with various applications including radial velocity fitting and transit modeling. For periodic VSs, we select a GP kernel based on a composition of SHOs with a quasi-periodic covariance term. For each SHO model, the associated power spectral density $S(\omega)$ is defined as follows:

$$S(\omega) = \sum_{j=1}^{N_{\rm SHO}} S_j(\omega), \tag{B1}$$

$$S_j(\omega) = \sqrt{\frac{2}{\pi}} \frac{S_{0,j}\,\omega_{0,j}^4}{(\omega^2 - \omega_{0,j}^2)^2 + \omega^2\omega_{0,j}^2/Q_j^2}, \tag{B2}$$

where $\omega_{0,j}$ and Q_j, respectively, refer to the frequency of the undamped oscillator and the associated quality factor of the jth oscillator. The parameter $S_{0,j}$ is proportional to the resonance (i.e., $\omega = \omega_{0,j}$) power.

Our data reduction approach is a twofold process: first, we fit a GP model to the observed data $\boldsymbol{x}_{\rm obs}$, and second, we use the model to predict a representation of the time series over a reduced time frame $\boldsymbol{T}_{\rm red}$ along with the uncertainties from the GP fit.

For periodic VSs, we use a GP model corresponding to a mixture of $N_{\rm SHO} = 2$ SHOs. We based our selection on the applications of GP modeling to data showcasing periodicity patterns as transit light curves or stellar variables in Foreman-Mackey et al. (2017). The GP model with two SHOs is chosen in order to take into consideration the multi-periodicity often encountered in variables stars. The current parameterization and kernel type are proper to the current work with a data set of pulsating variables and eclipsing binaries.

In the model parameterization, the periodicity (due to pulsation or binarity) is captured by the resonance frequency, such that:

$$\omega_{0,j} = \frac{4\pi Q_j}{P_j + \sqrt{4Q_j^2 - 1}}, \tag{B3}$$

$$S_{0,j} = \frac{A_j}{\omega_{0,j}\, Q_j}. \tag{B4}$$

Here, P_j and A_j refer, respectively, to the period and amplitude of the variability per SHO model j. For each observed light curve, an independent GP model is fitted. A full description of the GP modeling can be found in Foreman-Mackey et al. (2017) and the available open-source code by Foreman-Mackey et al. (2019).

Using the MAP solution, model prediction is performed on a time frame $\boldsymbol{T}_{\rm red}$ sampled within the range of observed epochs $\boldsymbol{T}_{\rm obs}$. For unevenly sampled data, the GP predictions located in large time gaps are associated with a high uncertainty of the model. To prevent such limitation, we developed an approach to generate a random time range within the observed $\boldsymbol{T}_{\rm obs}$ outside significant time gaps. Using the unsupervised K-means algorithm applied to the time differences $\Delta\boldsymbol{T}_{\rm obs}$, observations are clustered based on their proximity in time. Significant time gaps $\boldsymbol{T}_{\rm gaps}$ are identified within the group of large time differences, and an optional rejection criterion is supplemented to refine the detected time gaps to span, at least, higher than n cycles of the primary period of the light curve. The reduced time frame $\boldsymbol{T}_{\rm red}$ is generated via random sampling of time values within the observed time range outside of the identified time gaps $\mathbf{T_{set}} = \{\mathbf{T_{obs}} \setminus \mathbf{T_{gaps}}\}$. For each detected subset j of clustered observations, time values are obtained either by randomly generating values within the range of $\boldsymbol{T}_{{\rm set},j}$ or by randomly selecting of values in $\boldsymbol{T}_{{\rm set},j}$ shifted by a random $\delta_t > 0$. The second approach generates a time frame $\boldsymbol{T}_{\rm red}$ close to the observed $\boldsymbol{T}_{\rm set}$.

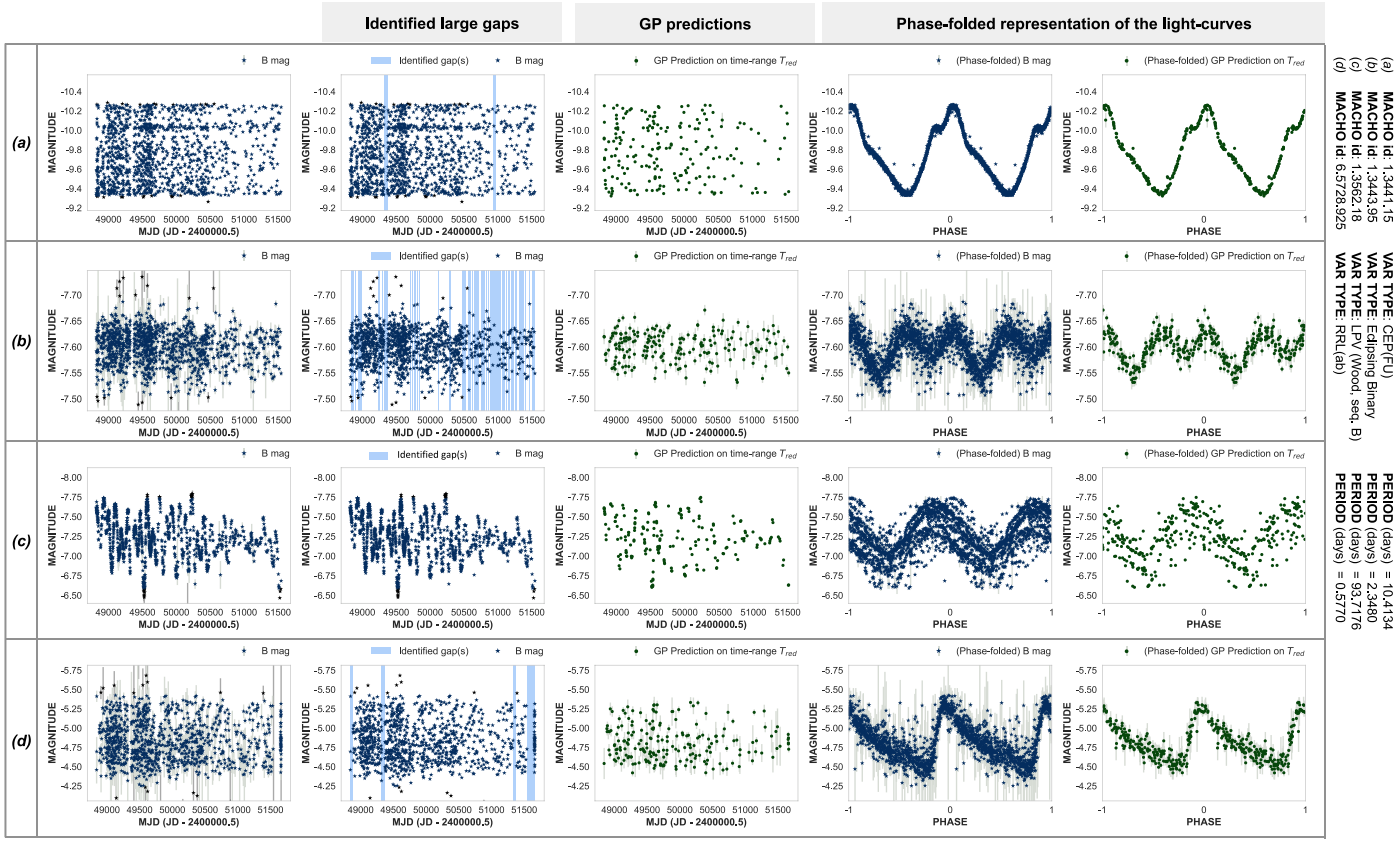

Figure B1. Displays of reduced MACHO light curves using GP model fit and prediction. A GP model is fitted to each observed light curve, and the best-fit model is associated with the MAP solution. The predictions—magnitudes and associated errors—are computed on a reduced time range $\boldsymbol{T}_{\rm red}$ of 200 data points obtained by a random selection of time values within the observed time range $\boldsymbol{T}_{\rm set}$ (outside the identified large gaps highlighted in light blue) and shifted with a random lag $\delta_t \in [0, \Delta T_{\rm set}]$. Phase-folded representations of light curves are given only as a reference. GP predictions are computed on the initial time series.

The GP model fitted to the data is a mixture of two SHOs with the following parameters:

$$
\begin{aligned}
&P_1 = P,\ P_2 = P/2,\ A_1 = \exp^{\log A},\ A_2 = m_A \times \exp^{\log A},\\
&Q_1 = Q_0 + \Delta Q,\ Q_2 = Q_0,\\
&m_A \sim \mathcal{U}(0,\,1),\ \log A \sim \mathcal{N}(\mu_A,\,\sigma_A^2),\\
&Q_0 > \frac{1}{2},\ \Delta Q \sim \mathcal{N}(\mu_Q,\,\sigma_Q^2),
\end{aligned}
\tag{B5}
$$

where P is the primary period of the time series, $(\mu_A,\,\sigma_A^2)$ refers to the amplitude of the time series and the associated error (or a fixed variance), and $(\mu_Q,\,\sigma_Q^2)$ are strictly positive values set to separate the two oscillation modes.

To illustrate the results of data reduction using the aforementioned GP model and prediction scheme, a display of MACHO light curves is provided in Figure B1.

Appendix C
Metrics for Multi-class Classification

To quantify the performances in multi-class classification, the precision, recall, F1-score, and accuracy are computed (see Table C1).

Table C1
Classification Metrics

Metrics	Per Class $\{j\}_{j:1\rightarrow N_C}$	Macro-averaging
Precision	$\text{Precision}_{(j)} = \frac{\text{TP}_{(j)}}{\text{TP}_{(j)} + \text{FP}_{(j)}}$	$\text{Precision}_M = \frac{1}{N_C}\sum_{j=1}^{N_C} \text{Precision}_{(j)}$
Recall (Sensitivity)	$\text{Recall}_{(j)} = \frac{\text{TP}_{(j)}}{\text{TP}_{(j)} + \text{FN}_{(j)}}$	$\text{Recall}_M = \frac{1}{N_C}\sum_{j=1}^{N_C} \text{Recall}_{(j)}$
F1-score	$\text{F1-score}_{(j)} = 2 \times \frac{\text{Recall}_{(j)} \times \text{Precision}_{(j)}}{\text{Recall}_{(j)} + \text{Precision}_{(j)}}$	$\text{F1-score}_M = \frac{1}{N_C}\sum_{j=1}^{N_C} \text{F1-score}_{(j)}$
Accuracy	$\text{Accuracy} = \frac{1}{N_s}\sum_{j=1}^{N_C} \text{TP}_{(j)}$	

Note. Where N_C is the total number of classes, N_s is the number oftrue samples, and $\text{TP}_{(j)}$, $\text{TN}_{(j)}$, $\text{FP}_{(j)}$, and $\text{FN}_{(j)}$, respectively, refer to the true positives, the true negatives, the false positives, and the false negatives computed from the predictions onobjects from the true class j.

Appendix D
Networks

This section presents visual representations and tables detailing the architectures of the neural networks used in this work. In particular, we reprensent in Figures D1–D3 the architectures of the direct classifiers and the composite networks. The encoder and decoder modules are constructed using stacks of different types of layers (RNNs, tCNNs, or dTCNs), while the classifier module is constructed using two dense layers.

The various configurations of the proposed architectures are described in Table D1 and D2, and we also report in Table D3 the network sizes, thus referring to the number of parameters in the networks.

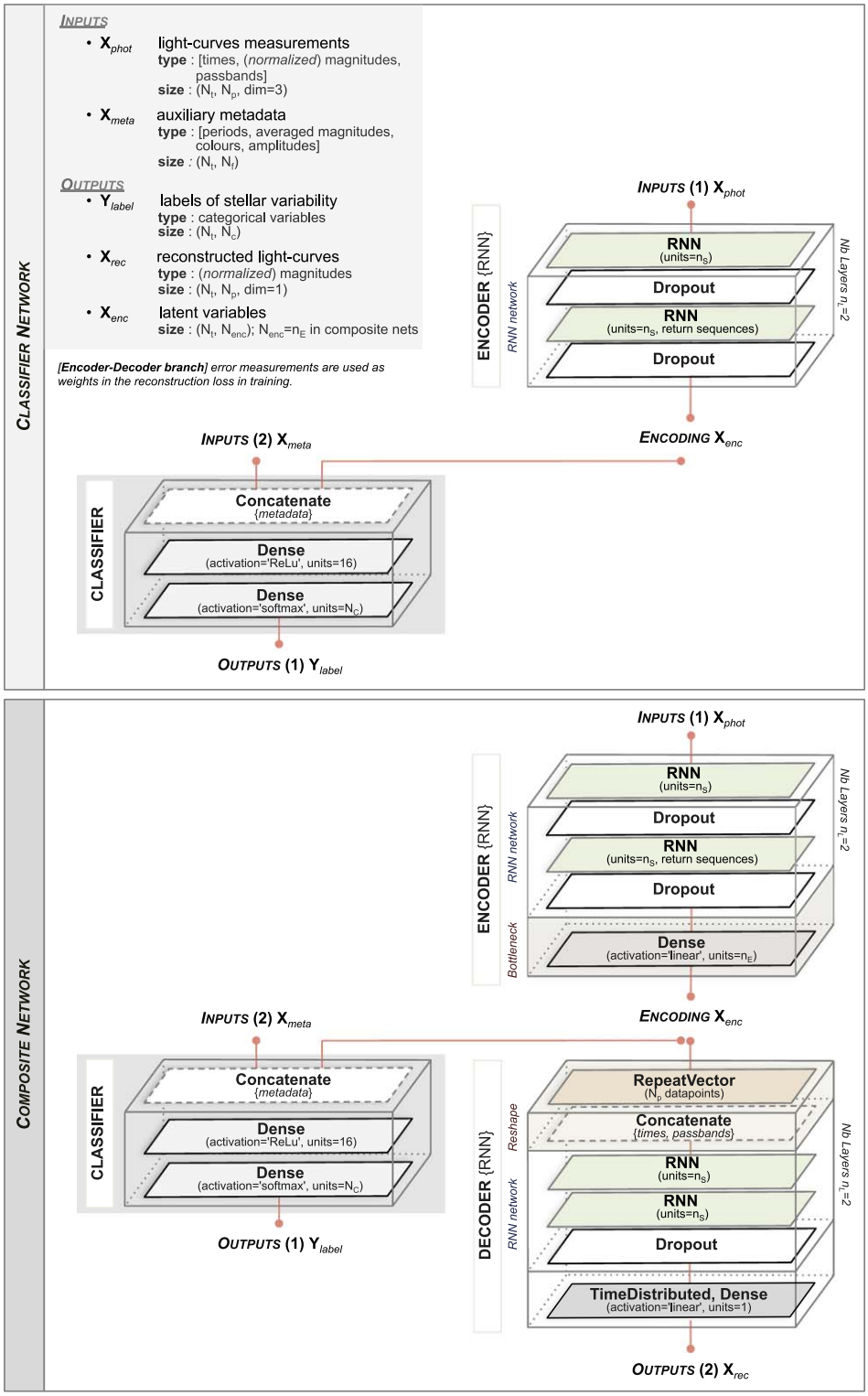

Figure D1. Architectures of the direct classifier and composite RNNs. Naming convention follows the implementation in `keras`.

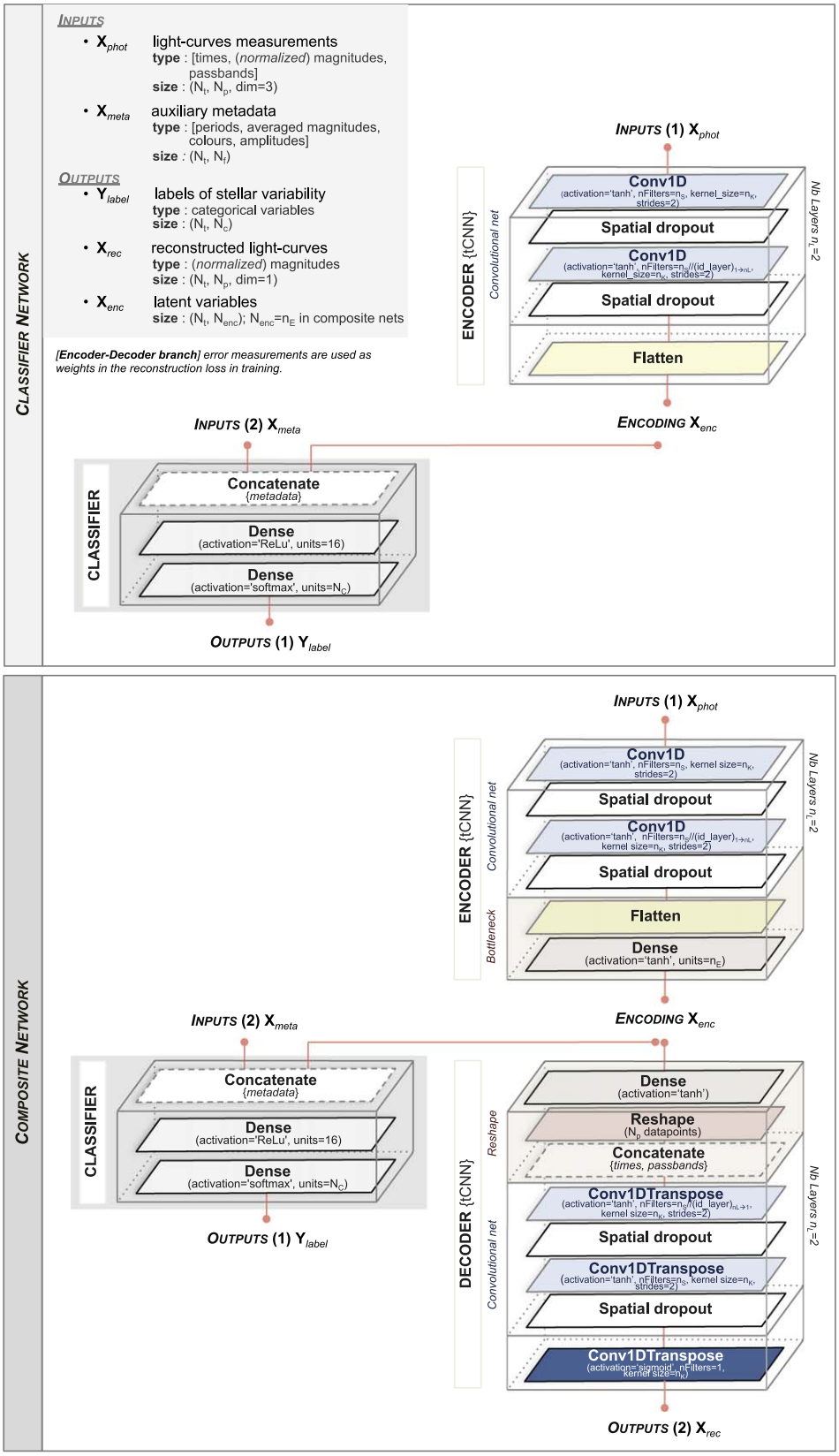

Figure D2. Architectures of the direct classifier and composite tCNNs. Naming convention follows the implementation in `keras`.

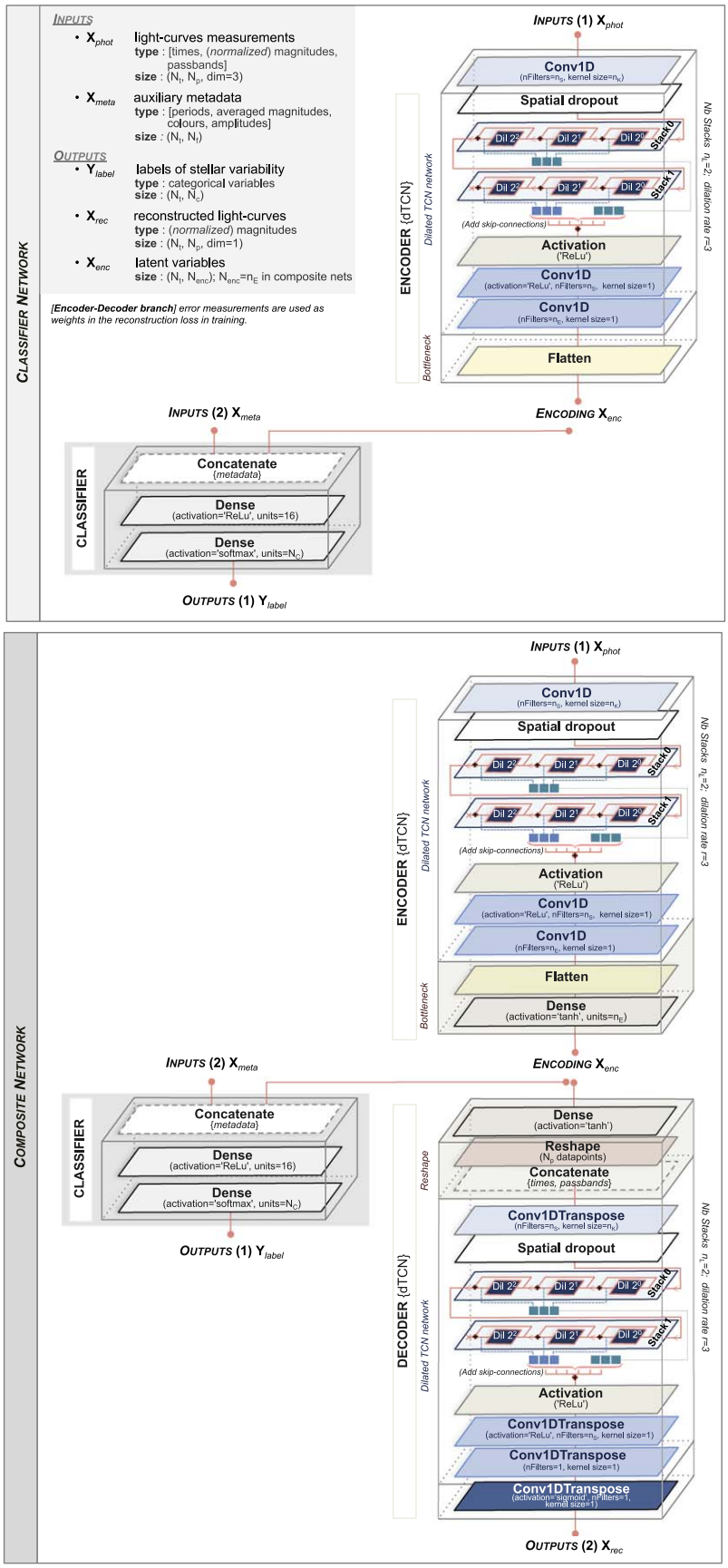

Figure D3. Architectures of the direct classifier and composite dTCNs. The series of dilated convolutions and residual stacks follow the `Wavenet` architecture (Oord et al. 2016) augmented with additional dropout functions to prevent overfitting. Naming convention follows the implementation in `keras`.

Table D1
Networks Configuration: Entries

Network type	Inputs	Outputs
Classifier c_F	$\boldsymbol{X}_{\rm phot}$	$\boldsymbol{Y}_{\rm label}$
Classifier $c_{F,\rm meta}$	$\{\boldsymbol{X}_{\rm phot}, \boldsymbol{X}_{\rm meta}\}$	$\boldsymbol{Y}_{\rm label}$
Composite d_F	$\boldsymbol{X}_{\rm phot}$	$\{\boldsymbol{Y}_{\rm label}, \boldsymbol{X}_{\rm rec}\}$
Composite $d_{F,\rm meta}$	$\{\boldsymbol{X}_{\rm phot}, \boldsymbol{X}_{\rm meta}\}$	$\{\boldsymbol{Y}_{\rm label}, \boldsymbol{X}_{\rm rec}\}$

Table D2
Networks Configuration: Hyperparameters

Layer Type	Configuration ID	Nb Layers (n_L)	Size (n_S)	Common Hyperparameters
RNN {LSTM; GRU}	*(1)*	1	16	Bidirectional network;
	(2)	2		Categorical classification;
	(3)	3		Drop fraction = 0.25;
	(4)	1	32	Batch size = 128; maximum training epochs = 200;
	(5)	2		`Adam`[a] optimizer; optimizer learning rate = 5×10^{-4};
	(6)	3		(Composite nets: embedding dimension $n_E = 8$; loss weights[b] = {1:1})
Layer Type	Configuration ID	Nb Layers (n_L)	Size (n_S)	Common Hyperparameters
Temporal CNN	*(1)*	1	16	Convolution kernel size $n_K = 5$;
	(2)	2		Categorical classification;
	(3)	3		Drop fraction = 0.25;
	(4)	1	32	Batch size = 128; maximum training epochs = 200;
	(5)	2		`Adam`[a] optimizer; optimizer learning rate = 5×10^{-4};
	(6)	3		(Composite nets: embedding dimension $n_E = 8$; loss weights[b] = {1:1})
Layer Type	Configuration ID	Nb Stacks (n_L)	Size (n_S)	Common Hyperparameters
Dilated TCN	*(1)*	1	16	Dilation rate[c] $r = 2$; convolution kernel size $n_K = 3$;
	(2)	2		Categorical classification;
	(3)	3		Drop fraction = 0.25;
	(4)	1	32	Batch size = 128; maximum training epochs = 200;
	(5)	2		`Adam`[a] optimizer; optimizer learning rate = 5×10^{-4};
	(6)	3		(Composite nets: embedding dimension $n_E = 8$; loss weights[b] = {1:1})

Notes.
[a] `Adam` optimization algorithm (Kingma & Ba 2017).
[b] Loss weights report the contributions of individual branches $\{w_{\rm ec}, w_{\rm ed}\}$ in the composite networks, respectively, the encoder-classifier and encoder-decoder branches.
[c] Dilation factors in dTCN correspond to $\{2^{j-1}\}_{j:1\rightarrow r}$, where $r > 1$ is the dilation rate.

Table D3
Network Sizes Corresponding to the Total Number of Parameters per Model across Three Different Data Sets (*B* band Only (Top), and Two Variants of the Combination of *R* and *B* Bands (Middle and Bottom))

MACHO – $B_{\rm band}$	Network Type	c_F	$c_{F,\rm meta}$	d_F	$d_{F,\rm meta}$
	LSTM	3292; *(1)*	3388; *(1)*	6661; *(1)*	6757; *(1)*
		9564; *(2)*	9660; *(2)*	19205; *(2)*	19301; *(2)*
		10460; *(4)*	10556; *(4)*	21157; *(4)*	21253; *(4)*
		15836; *(3)*	15932; *(3)*	31749; *(3)*	31845; *(3)*
		35292; *(5)*	35388; *(5)*	70821; *(5)*	70917; *(5)*
		60124; *(6)*	60220; *(6)*	120485; *(6)*	120581; *(6)*
	GRU	2652; *(1)*	2748; *(1)*	5157; *(1)*	5253; *(1)*
		7356; *(2)*	7452; *(2)*	14565; *(2)*	14661; *(2)*
		8156; *(4)*	8252; *(4)*	16101; *(4)*	16197; *(4)*
		12060; *(3)*	12156; *(3)*	23973; *(3)*	24069; *(3)*
		26780; *(5)*	26876; *(5)*	53349; *(5)*	53445; *(5)*
		45404; *(6)*	45500; *(6)*	90597; *(6)*	90693; *(6)*
	tCNN	5329; *(3)*	5425; *(3)*	8371; *(3)*	8467; *(3)*
		12118; *(6)*	12214; *(6)*	15925; *(2)*	16021; *(2)*
		13924; *(2)*	14020; *(2)*	16465; *(6)*	16561; *(6)*
		28908; *(5)*	29004; *(5)*	26853; *(5)*	26949; *(5)*
		51676; *(1)*	51772; *(1)*	41349; *(1)*	41445; *(1)*
		103132; *(4)*	103228; *(4)*	67941; *(4)*	68037; *(4)*
	dTCN	54100; *(1)*	54196; *(1)*	60335; *(1)*	60431; *(1)*
		56212; *(2)*	56308; *(2)*	64559; *(2)*	64655; *(2)*
		58324; *(3)*	58420; *(3)*	68783; *(3)*	68879; *(3)*
		61380; *(4)*	61476; *(4)*	75119; *(4)*	75215; *(4)*
		69700; *(5)*	69796; *(5)*	91759; *(5)*	91855; *(5)*
		78020; *(6)*	78116; *(6)*	108399; *(6)*	108495; *(6)*

MACHO – $RB_{\rm merged}$	Network Type	c_F	$c_{F,\rm meta}$	d_F	$d_{F,\rm meta}$
	LSTM	3292 ; *(1)*	3388 ; *(1)*	6661 ; *(1)*	6757 ; *(1)*
		9564 ; *(2)*	9660 ; *(2)*	19205 ; *(2)*	19301 ; *(2)*
		10460 ; *(4)*	10556 ; *(4)*	21157 ; *(4)*	21253 ; *(4)*
		15836 ; *(3)*	15932 ; *(3)*	31749 ; *(3)*	31845 ; *(3)*
		35292 ; *(5)*	35388 ; *(5)*	70821 ; *(5)*	70917 ; *(5)*
		60124 ; *(6)*	60220 ; *(6)*	120485 ; *(6)*	120581 ; *(6)*
	GRU	2652 ; *(1)*	2748 ; *(1)*	5157 ; *(1)*	5253 ; *(1)*
		7356 ; *(2)*	7452 ; *(2)*	14565 ; *(2)*	14661 ; *(2)*
		8156 ; *(4)*	8252 ; *(4)*	23973 ; *(3)*	16197 ; *(4)*
		12060 ; *(3)*	12156 ; *(3)*	16101 ; *(4)*	24069 ; *(3)*
		26780 ; *(5)*	26876 ; *(5)*	53349 ; *(5)*	53445 ; *(5)*
		45404 ; *(6)*	45500 ; *(6)*	90597 ; *(6)*	90693 ; *(6)*
	tCNN	9329 ; *(3)*	9425 ; *(3)*	13971 ; *(3)*	14067 ; *(3)*
		20118 ; *(6)*	20214 ; *(6)*	24065 ; *(6)*	24161 ; *(6)*

Table D3
(Continued)

MACHO – $B_{\rm band}$	Network Type	c_F	$c_{F,\rm meta}$	d_F	$d_{F,\rm meta}$
		26724 ; *(2)*	26820 ; *(2)*	29525 ; *(2)*	29621 ; *(2)*
		54508 ; *(5)*	54604 ; *(5)*	46853 ; *(5)*	46949 ; *(5)*
		102876 ; *(1)*	102972 ; *(1)*	81349 ; *(1)*	81445 ; *(1)*
		205532 ; *(4)*	205628 ; *(4)*	133541 ; *(4)*	133637 ; *(4)*
	dTCN	105300 ; *(1)*	105396 ; *(1)*	114735 ; *(1)*	114831 ; *(1)*
		107412 ; *(2)*	107508 ; *(2)*	118959 ; *(2)*	119055 ; *(2)*
		109524 ; *(3)*	109620 ; *(3)*	123183 ; *(3)*	123279 ; *(3)*
		112580 ; *(4)*	112676 ; *(4)*	129519 ; *(4)*	129615 ; *(4)*
		120900 ; *(5)*	120996 ; *(5)*	146159 ; *(5)*	146255 ; *(5)*
		129220 ; *(6)*	129316 ; *(6)*	162799 ; *(6)*	162895 ; *(6)*

MACHO – $RB_{\rm hybrid}$	Network Type	c_F	$c_{F,\rm meta}$	d_F	$d_{F,\rm meta}$
	LSTM	6364 ; *(1)*	6460 ; *(1)*	13110 ; *(1)*	13206 ; *(1)*
		18908 ; *(2)*	19004 ; *(2)*	38198 ; *(2)*	38294 ; *(2)*
		20700 ; *(4)*	20796 ; *(4)*	42102 ; *(4)*	42198 ; *(4)*
		31452 ; *(3)*	31548 ; *(3)*	63286 ; *(3)*	63382 ; *(3)*
		70364 ; *(5)*	70460 ; *(5)*	141430 ; *(5)*	141526 ; *(5)*
		120028 ; *(6)*	120124 ; *(6)*	240758 ; *(6)*	240854 ; *(6)*
	GRU	5084 ; *(1)*	5180 ; *(1)*	10102 ; *(1)*	10198 ; *(1)*
		14492 ; *(2)*	14588 ; *(2)*	28918 ; *(2)*	29014 ; *(2)*
		16092 ; *(4)*	16188 ; *(4)*	31990 ; *(4)*	32086 ; *(4)*
		23900 ; *(3)*	23996 ; *(3)*	47734 ; *(3)*	47830 ; *(3)*
		53340 ; *(5)*	53436 ; *(5)*	106486 ; *(5)*	106582 ; *(5)*
		90588 ; *(6)*	90684 ; *(6)*	180982 ; *(6)*	181078 ; *(6)*
	tCNN	10438 ; *(3)*	10534 ; *(3)*	16530 ; *(3)*	16626 ; *(3)*
		24016 ; *(6)*	24112 ; *(6)*	31638 ; *(2)*	31734 ; *(2)*
		27628 ; *(2)*	27724 ; *(2)*	32718 ; *(6)*	32814 ; *(6)*
		57596 ; *(5)*	57692 ; *(5)*	53494 ; *(5)*	53590 ; *(5)*
		103132 ; *(1)*	103228 ; *(1)*	82486 ; *(1)*	82582 ; *(1)*
		206044 ; *(4)*	206140 ; *(4)*	135670 ; *(4)*	135766 ; *(4)*
	dTCN	107980 ; *(1)*	108076 ; *(1)*	120458 ; *(1)*	120554 ; *(1)*
		112204 ; *(2)*	112300 ; *(2)*	128906 ; *(2)*	129002 ; *(2)*
		116428 ; *(3)*	116524 ; *(3)*	137354 ; *(3)*	137450 ; *(3)*
		122540 ; *(4)*	122636 ; *(4)*	150026 ; *(4)*	150122 ; *(4)*
		139180 ; *(5)*	139276 ; *(5)*	183306 ; *(5)*	183402 ; *(5)*
		155820 ; *(6)*	155916 ; *(6)*	216586 ; *(6)*	216682 ; *(6)*

Note. The identifiers of the hyperparameter set configurations *(1)* to *(6)* are ordered by increasing size. The largest networks for the RNNs, tCNNs, and dTCNs correspond, respectively, to the identifiers *(6)*, *(4)*, and *(6)*, and the smallest configurations correspond, respectively, to *(1)*, *(3)*, and *(1)*. The metadata in the current application ($N_f = 6$) adds 96 parameters.

Appendix E
Global Performances

This section summarizes the main results of the VS classification tests on the MACHO data with the neural networks architectures presented in this work.

We provide results obtained for all tested configurations across the three different data sets (*B* band and the two variants of *R* and *B* bands).

The training convergence times are detailed in Figure E1, while Figures E2 and E3 report on the total loss and accuracy values obtained by the networks at convergence (i.e., the end of training) for the training, validation, and test data sets. Moreover, the evolution of the metrics (total loss and accuracy) during training is illustrated in Figures E4 and E5 for the LSTM composite networks d_F and $d_{F,meta}$.

For the best performing composite networks using the *B* band, examples of reconstructed light curves are illustrated in Figure E6 and E7. The distribution of the reconstruction errors is further reported on Figure E8.

In this section, the representation of the generated embeddings is reported on Figures E9–E12 for the best performing LSTM direct classifier and composite networks.

The identifiers of the best performing networks in the current tests are summarized on Table E1.

Detailed results on the accuracy and classification metrics are reported for the best-performing networks in Tables E2–E5.

		LSTM						GRU						tCNN						dTCN					
		(1)	*(2)*	*(3)*	*(4)*	*(5)*	*(6)*	*(1)*	*(2)*	*(3)*	*(4)*	*(5)*	*(6)*	*(1)*	*(2)*	*(3)*	*(4)*	*(5)*	*(6)*	*(1)*	*(2)*	*(3)*	*(4)*	*(5)*	*(6)*
MACHO - B_{band}	c_F	1.04	2.59	2.64	0.24	2.75	4.25	0.83	2.27	2.28	0.78	2.67	4.42	0.02	0.04	0.09	0.03	0.05	0.11	0.09	0.15	0.24	0.15	0.25	0.37
	$c_{F,meta}$	0.78	2.08	3.45	1.00	3.12	5.19	0.62	1.84	2.42	0.83	2.75	4.02	0.06	0.08	0.09	0.04	0.06	0.14	0.17	0.16	0.27	0.17	0.31	0.48
	d_F	2.24	5.51	7.05	2.42	7.78	6.06	1.54	4.59	5.79	2.51	6.28	8.54	0.08	0.16	0.19	0.23	0.35	0.23	0.59	0.66	0.75	0.99	1.27	1.58
	$d_{F,meta}$	1.80	5.56	8.60	2.91	7.94	12.76	1.50	3.72	7.78	1.99	4.79	8.55	0.10	0.13	0.22	0.23	0.32	0.45	0.42	0.86	0.90	0.83	2.91	4.61
MACHO-RB_{merged}	c_F	0.58	1.76	7.62	0.69	2.55	9.45	1.53	3.61	1.47	1.02	4.05	9.24	0.02	0.04	0.15	0.02	0.05	0.09	0.11	0.16	0.27	0.24	0.33	0.46
	$c_{F,meta}$	1.28	4.18	8.14	2.00	6.04	8.06	1.57	4.40	6.98	1.76	6.65	9.02	0.02	0.04	0.16	0.02	0.05	0.08	0.12	0.19	0.27	0.21	0.37	0.64
	d_F	3.70	8.68	8.96	1.20	12.98	17.70	3.00	7.04	14.05	4.69	12.85	13.05	0.25	0.32	0.43	0.31	0.31	0.55	0.74	1.21	1.68	1.54	2.47	2.62
	$d_{F,meta}$	4.31	10.88	14.04	4.71	13.03	20.75	3.81	9.37	11.93	3.86	13.31	17.99	0.28	0.35	0.38	0.36	0.49	0.86	0.76	2.61	4.82	1.50	5.10	9.33
MACHO - RB_{hybrid}	c_F	0.36	1.92	2.09	0.58	4.53	6.31	0.98	2.82	4.70	1.46	3.29	6.01	0.03	0.04	0.11	0.04	0.07	0.12	0.15	0.24	0.26	0.26	0.38	0.52
	$c_{F,meta}$	1.28	3.37	5.79	1.73	5.41	9.26	1.08	3.26	5.39	1.42	5.02	7.63	0.04	0.05	0.15	0.06	0.06	0.18	0.15	0.22	0.26	0.25	0.41	0.53
	d_F	2.94	6.55	11.40	4.51	12.40	13.07	2.94	6.16	9.30	3.28	9.63	8.82	0.17	0.15	0.34	0.17	0.36	0.40	0.47	0.53	1.05	2.36	2.43	1.87
	$d_{F,meta}$	3.17	3.80	12.65	4.43	11.23	12.74	2.38	5.59	8.51	3.57	9.65	17.65	0.17	0.12	0.49	0.22	0.33	0.28	0.59	0.88	1.87	1.15	1.36	2.24

Figure E1. Training convergence time (expressed in hours) of all models across three different data sets (*B*-band only (top), and two variants of the combination of *R* and *B* bands (middle and bottom)). Runs are performed on a CPU model `Intel Xeon E5-2643v3` using four cores per run at maximum capacity. The identifiers of the hyperparameter set configurations *(1)* to *(6)* are stated in Table D2.

MACHO - B_{band}

		LSTM						GRU						tCNN						dTCN					
		(1)	*(2)*	*(3)*	*(4)*	*(5)*	*(6)*	*(1)*	*(2)*	*(3)*	*(4)*	*(5)*	*(6)*	*(1)*	*(2)*	*(3)*	*(4)*	*(5)*	*(6)*	*(1)*	*(2)*	*(3)*	*(4)*	*(5)*	*(6)*
Train set	c_F	0.76	0.73	0.74	0.90	0.68	0.67	0.98	0.66	0.71	0.74	0.58	0.53	0.79	0.71	0.72	0.80	0.67	0.70	0.74	0.52	0.66	0.53	0.60	0.44
	$c_{F,meta}$	0.22	2.57	1.86	0.89	0.89	0.89	0.89	0.90	0.31	0.27	0.23	0.24	0.24	0.20	0.25	0.32	0.29	0.87	0.40	0.21	0.29	0.27	0.19	0.20
	d_F	8.08	7.83	7.85	7.62	7.55	7.60	8.17	8.16	7.96	7.69	7.55	7.52	7.72	7.65	8.03	7.54	7.69	7.97	7.29	7.31	7.49	7.15	7.91	7.36
	$d_{F,meta}$	8.99	7.87	7.91	9.01	7.64	6.88	8.82	8.19	9.47	7.77	7.78	7.84	8.87	7.12	7.32	8.72	7.02	7.12	7.64	8.00	7.94	7.67	15.50	10.50
Validation set	c_F	0.77	0.76	0.75	0.92	0.72	0.71	0.98	0.68	0.72	0.75	0.62	0.63	0.92	0.83	0.80	0.94	0.81	0.81	1.00	1.04	0.99	1.03	1.09	1.14
	$c_{F,meta}$	0.24	2.59	1.88	0.91	0.91	0.91	0.90	0.92	0.34	0.29	0.27	0.30	0.42	0.35	0.32	0.51	0.45	1.00	0.60	0.71	0.64	0.61	0.83	0.90
	d_F	7.89	7.75	7.79	7.54	7.50	7.58	7.98	8.06	7.80	7.54	7.48	7.49	7.80	7.67	7.92	7.61	7.73	7.90	7.59	7.45	7.63	7.44	8.10	7.59
	$d_{F,meta}$	8.87	7.79	7.85	8.89	7.60	6.86	8.67	8.06	9.33	7.69	7.71	7.81	8.92	7.12	7.26	8.82	7.04	7.07	7.77	8.18	8.09	7.80	15.13	10.37
Test set	c_F	0.80	0.78	0.78	0.95	0.76	0.74	1.02	0.73	0.79	0.79	0.68	0.65	0.93	0.85	0.82	0.96	0.86	0.83	1.03	1.07	1.06	1.10	1.12	1.21
	$c_{F,meta}$	0.25	2.60	1.90	0.93	0.94	0.94	0.92	0.95	0.36	0.30	0.28	0.33	0.43	0.39	0.35	0.51	0.46	1.00	0.60	0.70	0.65	0.62	0.89	0.92
	d_F	8.07	7.85	7.86	7.67	7.60	7.67	8.14	8.17	7.98	7.75	7.60	7.61	7.88	7.74	8.06	7.72	7.84	7.98	7.64	7.58	7.74	7.53	8.21	7.65
	$d_{F,meta}$	8.99	7.91	7.98	9.05	7.70	6.97	8.82	8.18	9.43	7.81	7.86	7.91	9.06	7.22	7.33	8.91	7.15	7.19	7.86	8.26	8.18	7.89	15.31	10.56

MACHO - RB_{merged}

		LSTM						GRU						tCNN						dTCN					
		(1)	*(2)*	*(3)*	*(4)*	*(5)*	*(6)*	*(1)*	*(2)*	*(3)*	*(4)*	*(5)*	*(6)*	*(1)*	*(2)*	*(3)*	*(4)*	*(5)*	*(6)*	*(1)*	*(2)*	*(3)*	*(4)*	*(5)*	*(6)*
Train set	c_F	1.20	1.01	0.75	1.22	0.72	0.68	0.76	0.66	1.10	0.77	0.60	0.56	0.59	0.56	0.64	0.72	0.57	0.61	0.44	0.43	0.43	0.42	0.32	0.13
	$c_{F,meta}$	0.32	2.58	1.88	0.91	0.89	0.94	0.90	0.89	0.26	0.33	0.22	0.23	0.22	0.26	0.85	0.29	0.22	0.35	0.20	0.17	0.12	0.15	0.10	0.04
	d_F	10.98	10.66	11.04	12.34	10.23	10.48	11.02	11.04	11.85	10.45	10.32	10.80	10.08	10.27	10.77	9.99	10.14	10.54	9.61	9.68	10.17	9.37	9.34	9.77
	$d_{F,meta}$	12.06	10.96	11.00	11.94	10.50	9.75	11.80	10.93	12.79	10.53	10.55	10.69	11.20	9.65	10.16	11.26	9.51	9.73	9.91	18.73	18.72	9.77	18.66	18.59
Validation set	c_F	1.19	1.01	0.75	1.23	0.74	0.71	0.76	0.68	1.09	0.76	0.64	0.62	1.16	0.92	0.82	1.15	0.94	0.81	1.27	1.39	1.21	1.34	1.53	2.18
	$c_{F,meta}$	0.32	2.59	1.91	0.93	0.91	0.94	0.91	0.91	0.30	0.34	0.25	0.29	0.81	0.51	1.01	0.78	0.62	0.52	0.95	1.02	1.17	1.05	1.13	1.54
	d_F	10.44	10.14	10.51	11.62	9.91	10.13	10.44	10.48	11.27	9.97	9.88	10.32	10.47	10.24	10.45	10.37	10.24	10.36	10.16	10.12	10.41	10.03	10.02	10.13
	$d_{F,meta}$	11.48	10.42	10.50	11.46	10.18	9.43	11.26	10.45	12.26	10.04	10.18	10.27	11.52	9.65	9.84	11.59	9.61	9.61	10.30	17.91	17.91	10.16	17.89	17.86
Test set	c_F	1.21	1.04	0.80	1.24	0.77	0.75	0.79	0.73	1.15	0.79	0.69	0.67	1.23	0.97	0.87	1.21	1.00	0.87	1.38	1.46	1.26	1.44	1.60	2.36
	$c_{F,meta}$	0.34	2.61	1.92	0.94	0.94	0.96	0.93	0.94	0.31	0.37	0.27	0.31	0.81	0.55	1.05	0.80	0.65	0.55	0.99	1.10	1.24	1.10	1.19	1.58
	d_F	10.86	10.57	11.06	12.21	10.29	10.54	10.90	10.94	11.73	10.39	10.27	10.74	10.80	10.59	10.78	10.71	10.55	10.71	10.41	10.41	10.75	10.35	10.28	10.45
	$d_{F,meta}$	11.96	10.90	11.00	11.89	10.53	9.79	11.69	10.86	12.69	10.48	10.55	10.64	11.85	10.00	10.10	11.93	9.92	9.92	10.57	18.14	18.15	10.46	18.10	18.11

MACHO - RB_{hybrid}

		LSTM						GRU						tCNN						dTCN					
		(1)	*(2)*	*(3)*	*(4)*	*(5)*	*(6)*	*(1)*	*(2)*	*(3)*	*(4)*	*(5)*	*(6)*	*(1)*	*(2)*	*(3)*	*(4)*	*(5)*	*(6)*	*(1)*	*(2)*	*(3)*	*(4)*	*(5)*	*(6)*
Train set	c_F	1.15	0.70	0.78	0.88	0.55	0.60	0.69	0.63	0.57	0.60	0.54	0.48	0.69	0.61	0.62	0.87	0.63	0.56	0.53	0.48	0.34	0.47	0.27	0.37
	$c_{F,meta}$	0.90	0.92	0.23	1.17	0.27	0.90	0.94	0.24	0.90	0.29	0.87	0.90	0.29	0.26	0.92	0.37	0.33	0.26	0.17	0.11	0.15	0.13	0.06	0.07
	d_F	19.88	19.79	19.09	18.57	18.41	18.60	19.79	19.85	19.96	18.69	18.21	19.16	18.32	18.90	19.66	18.69	18.43	18.85	18.08	18.41	18.34	16.86	17.28	18.39
	$d_{F,meta}$	20.10	22.06	20.15	20.34	17.95	19.69	19.72	21.92	21.18	19.76	20.12	19.77	17.99	19.25	19.01	17.75	18.61	19.23	18.45	19.92	26.67	17.99	18.79	29.38
Validation set	c_F	1.17	0.70	0.79	0.89	0.62	0.63	0.70	0.65	0.61	0.63	0.61	0.59	0.91	0.77	0.74	1.00	0.85	0.76	1.00	0.98	1.16	1.11	1.32	1.22
	$c_{F,meta}$	0.92	0.93	0.26	1.20	0.30	0.93	0.95	0.27	0.94	0.30	0.92	1.00	0.52	0.48	1.05	0.60	0.53	0.45	0.79	0.82	0.92	0.94	1.14	1.20
	d_F	18.72	18.65	18.15	17.61	17.59	17.82	18.66	18.84	18.97	17.68	17.41	18.19	17.92	18.13	18.68	18.21	17.91	17.96	17.85	17.93	17.96	17.11	17.35	18.08
	$d_{F,meta}$	18.94	20.71	19.26	19.41	17.17	18.89	18.59	20.89	20.13	18.79	19.30	18.91	17.58	18.49	18.12	17.38	18.05	18.38	18.20	19.60	25.99	17.93	18.64	28.26
Test set	c_F	1.20	0.75	0.83	0.94	0.68	0.69	0.73	0.70	0.65	0.67	0.67	0.66	0.94	0.82	0.79	1.01	0.89	0.80	1.04	1.04	1.26	1.11	1.40	1.25
	$c_{F,meta}$	0.93	0.97	0.27	1.21	0.31	0.95	0.98	0.29	0.97	0.32	0.94	1.04	0.56	0.48	1.07	0.62	0.54	0.48	0.81	0.87	0.98	0.91	1.07	1.21
	d_F	19.49	19.60	18.89	18.36	18.30	18.51	19.45	19.55	19.72	18.52	18.12	18.99	18.65	18.86	19.34	18.98	18.53	18.70	18.47	18.65	18.66	17.78	18.03	18.75
	$d_{F,meta}$	19.74	21.72	19.98	20.11	17.83	19.59	19.36	21.62	20.92	19.59	20.00	19.72	18.32	19.22	18.85	18.06	18.74	19.00	18.95	20.36	26.67	18.56	19.19	28.71

Figure E2. Total loss (weighted MAE for reconstruction and categorical cross-entropy for classification) computed for all trained models across three different data sets (*B*-band only (top), and two variants of the combination of *R* and *B* bands (middle and bottom)). Solid underlines highlight the best-performing models associated with the minimum loss obtained in the test set. The identifiers *(1)* to *(6)* refer to the hyperparameter set configurations.

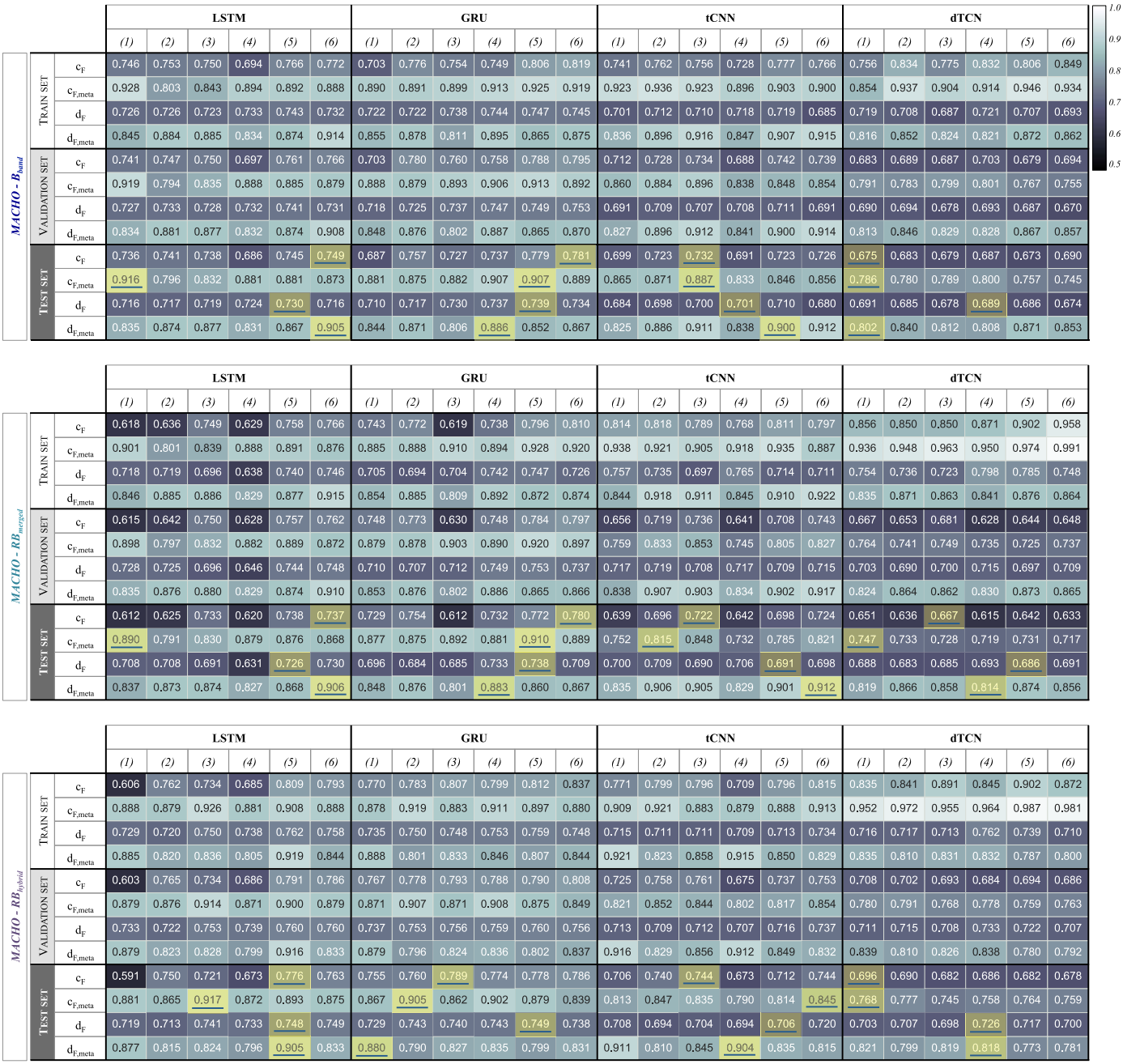

Figure E3. Classification accuracy, defined as the fraction of true positives within the sample, computed for all trained models across three different data sets (*B*-band only (top), and two variants of the combination of *R* and *B* bands (middle and bottom)). Solid underlines highlight the best-performing models associated with the minimum loss obtained in the test set. The identifiers *(1)* to *(6)* refer to the hyperparameter set configurations.

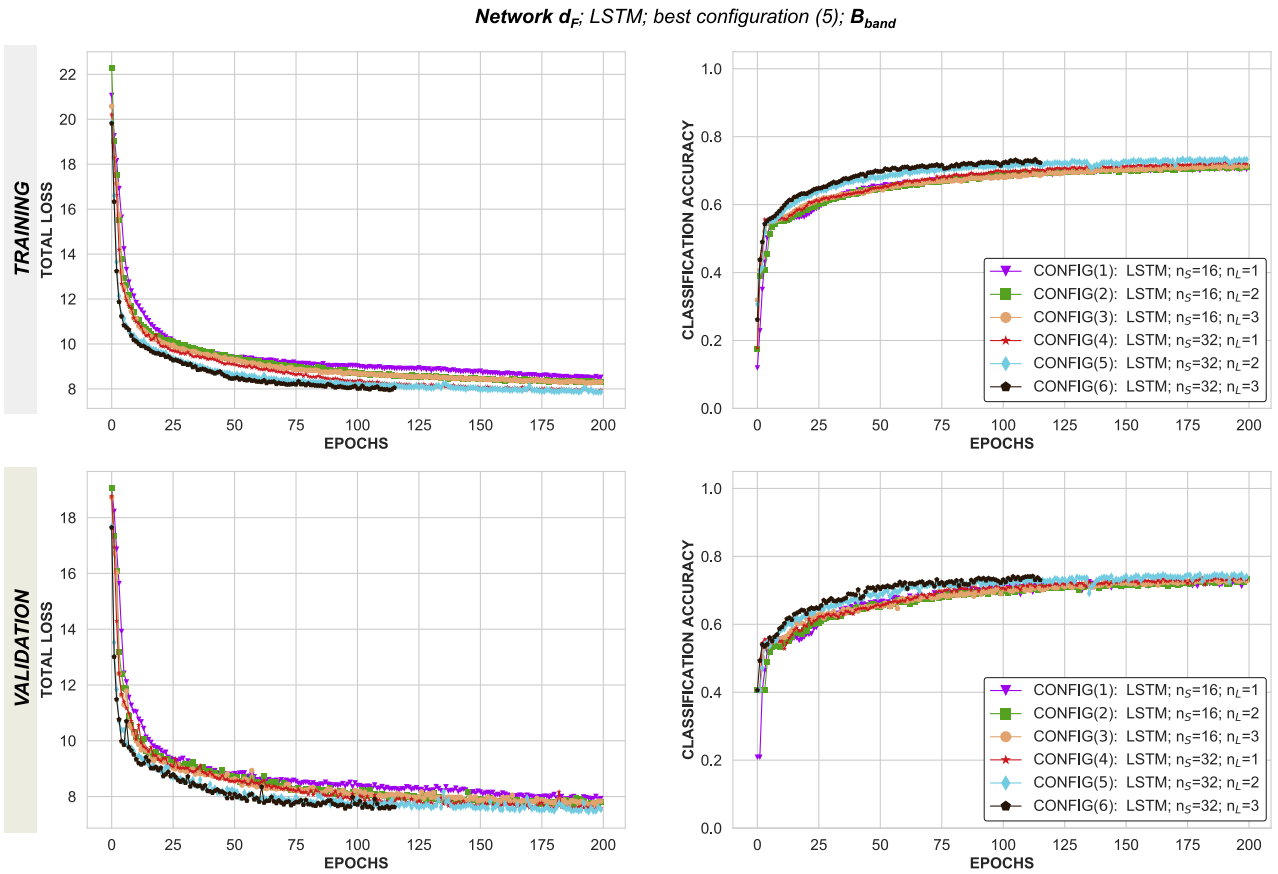

Figure E4. Total loss and classification accuracy for the LSTM composite d_F on the B band.

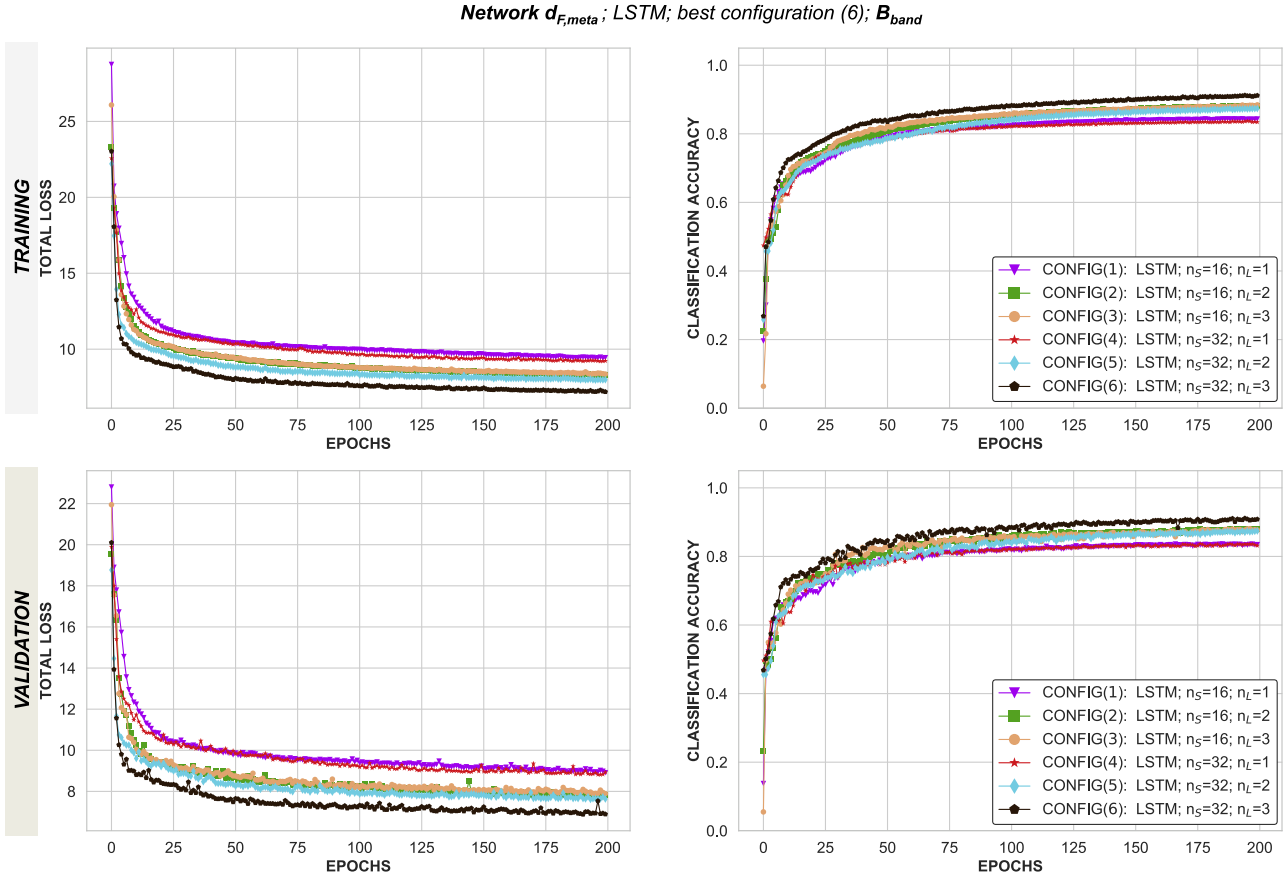

Figure E5. Total loss and classification accuracy for the LSTM composite $d_{F,\mathrm{meta}}$ using metadata as a secondary input on the B band.

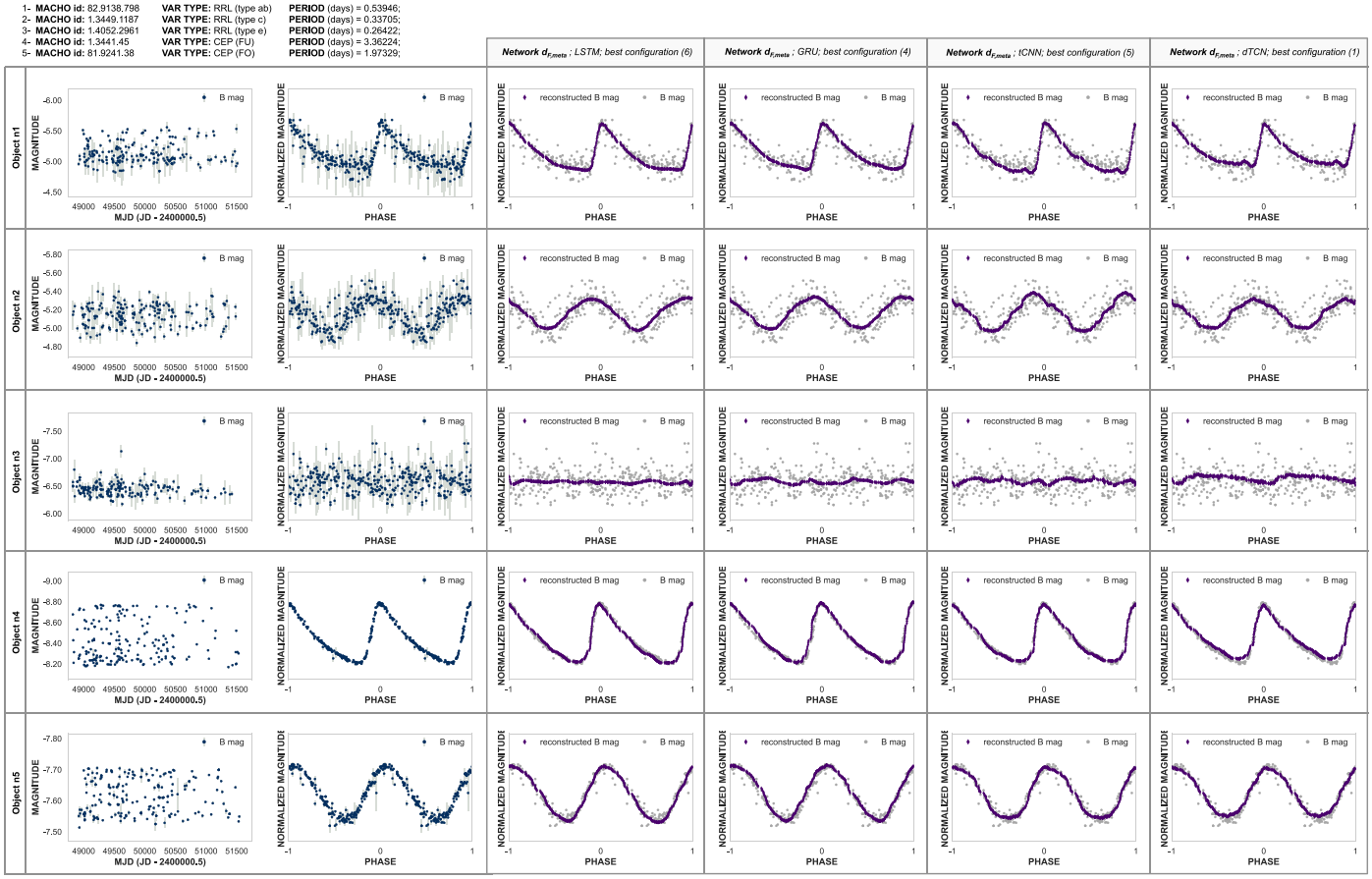

Figure E6. Displays (1) of reconstructed light curves from the test set for the best-performing composite $d_{F,\mathrm{meta}}$ on the B band (left to right: the best-performing LSTM, GRU, tCNN, and dTCN). For visualization purposes, the 1σ error measurements of the input data are not displayed in the reconstruction results.

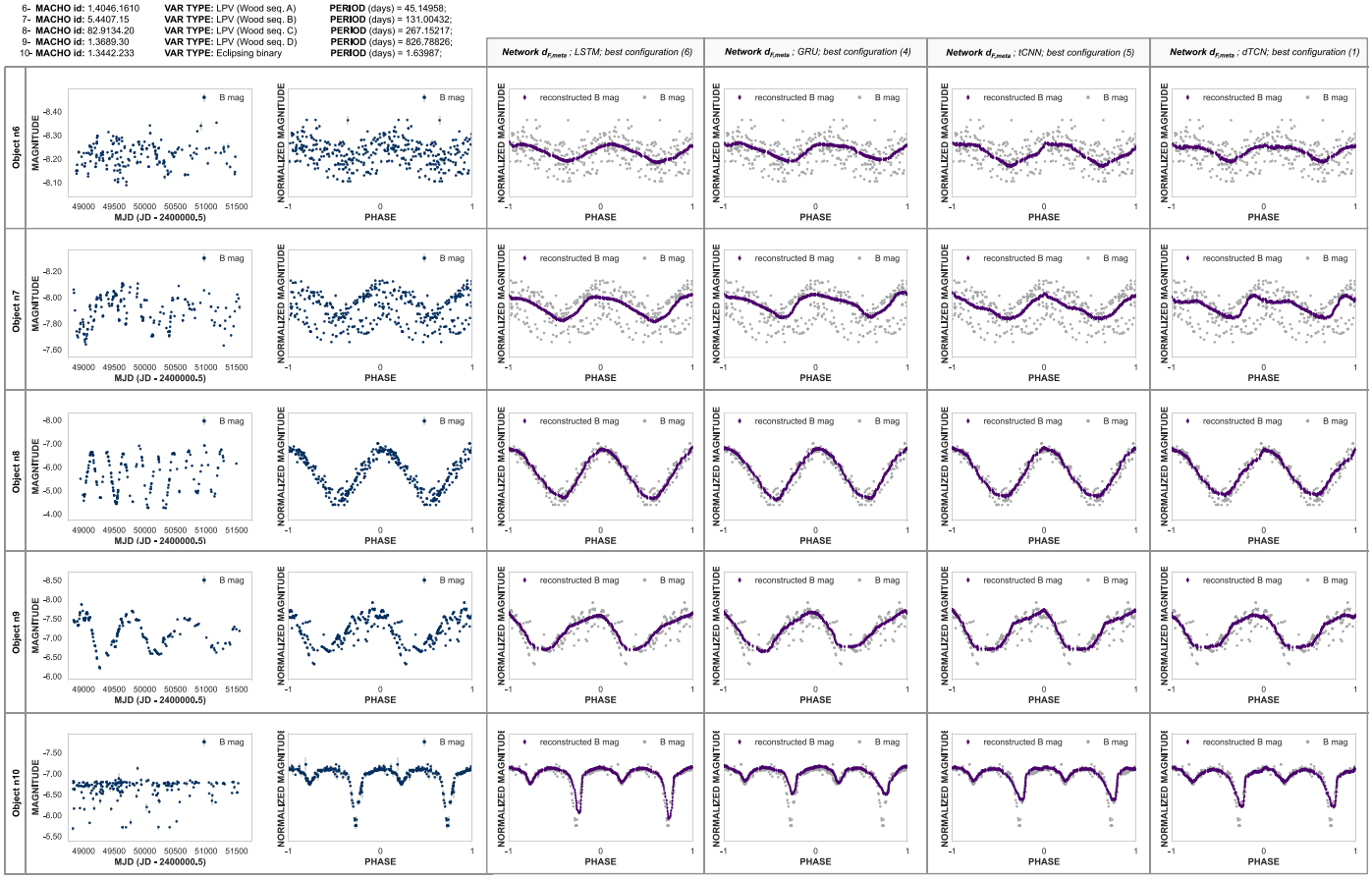

Figure E7. Displays (2) of reconstructed light curves from the test set for the best-performing composite $d_{F,\mathrm{meta}}$ on the B band (left to right: the best-performing LSTM, GRU, tCNN, and dTCN). For visualization purposes, the 1σ error measurements of the input data are not displayed in the reconstruction results.

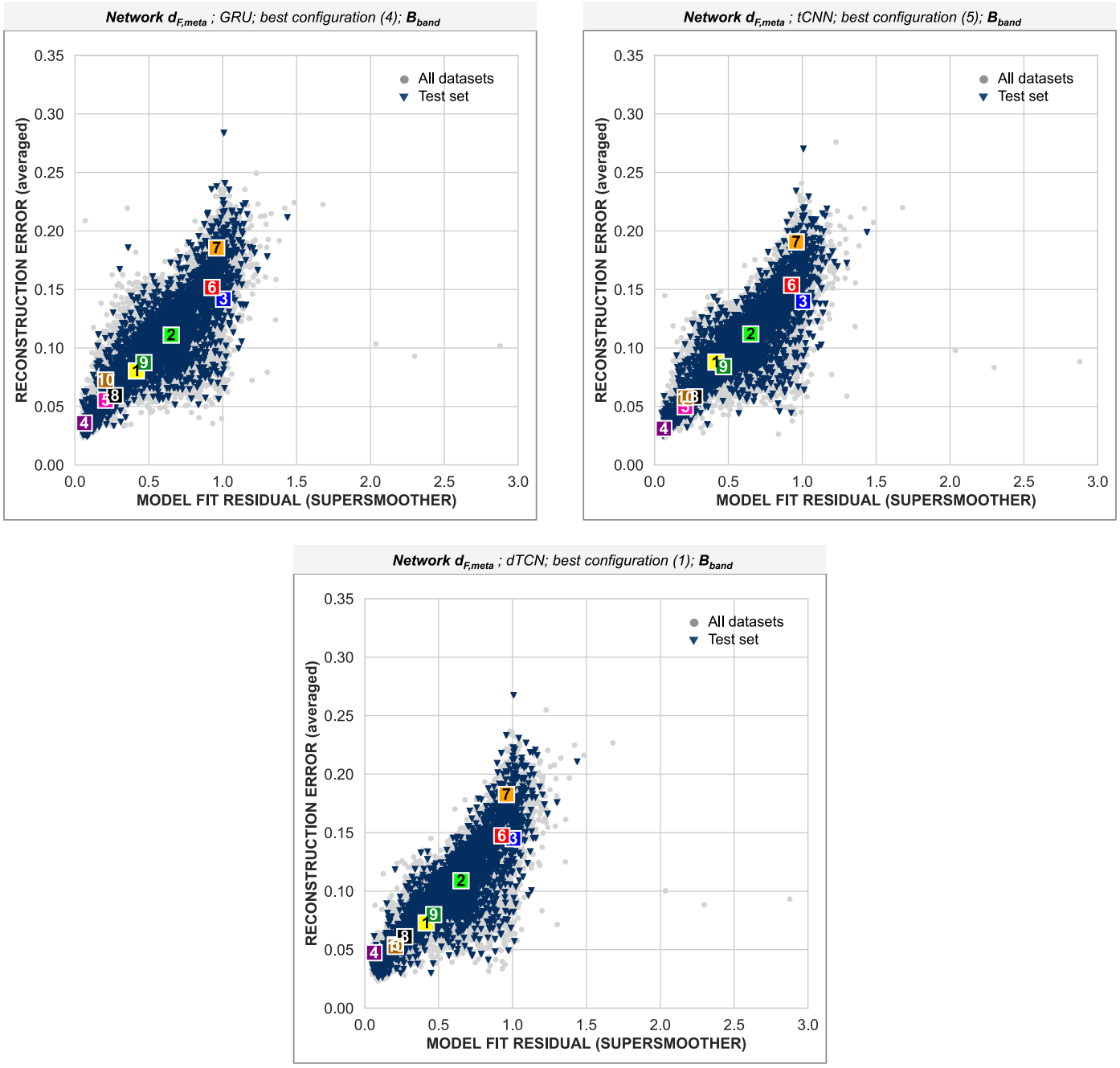

Figure E8. Reconstruction error (MAE) as a function of the model-fit residuals from the `SuperSmoother` algorithm (Friedman 1984) for the best-performing GRU, tCNN (top, left to right), and dTCN (bottom) $d_{F,\mathrm{meta}}$ on the B band. The highlighted numbers (1 to 10) refer to the subset of selected objects from the test set used to showcase the reconstruction quality of the autoencoder branch.

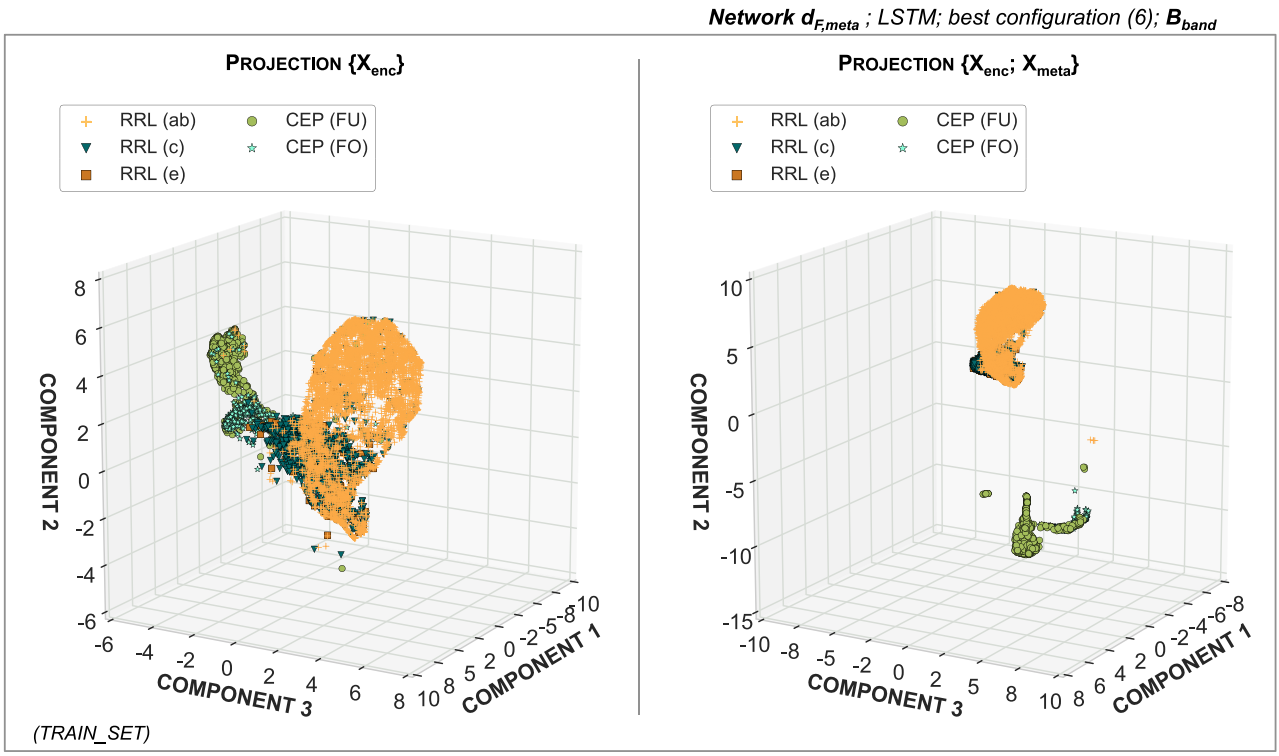

Figure E9. Three-dimensional representation of encoded features for the short-period pulsators in MACHO. The generated features from the best-performing LSTM composite network $d_{F,\mathrm{meta}}$ on the B band are projected onto a reduced 3D representation using the UMAP algorithm.

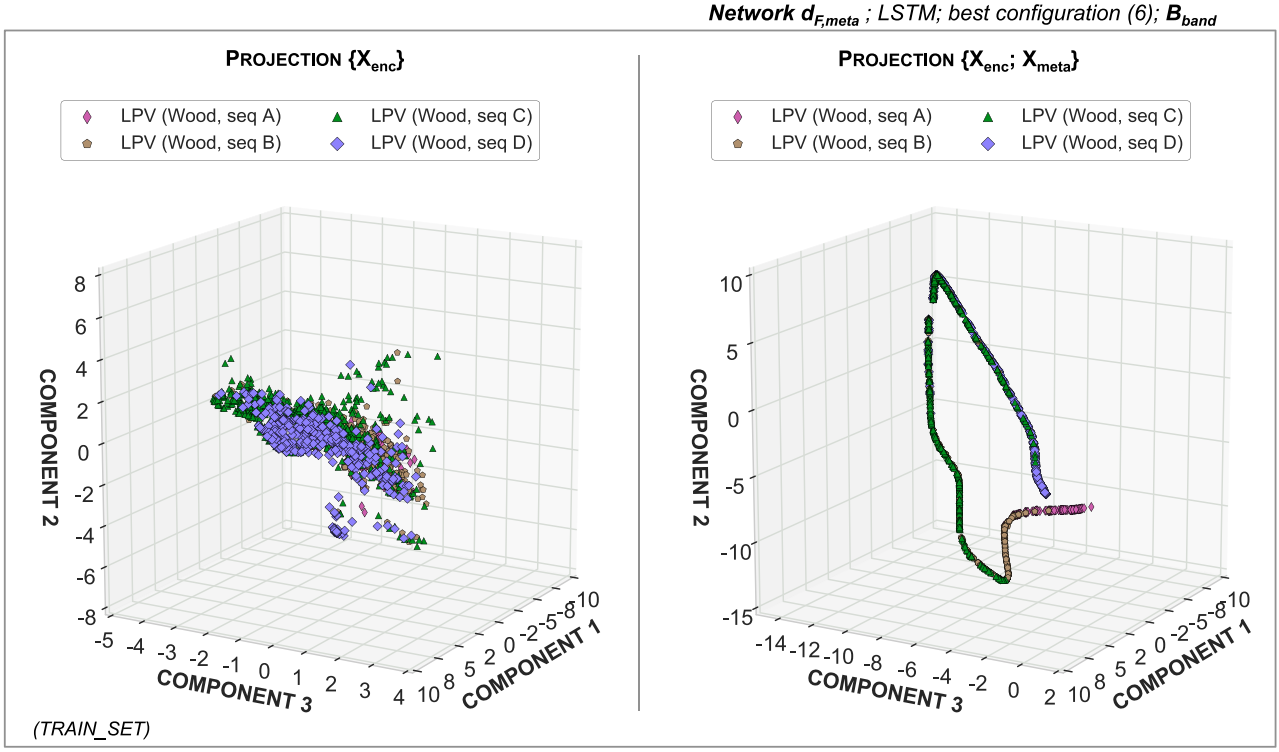

Figure E10. Three-dimensional representation of encoded features for the LPVs in MACHO. The generated features from the best-performing LSTM composite network $d_{F,\mathrm{meta}}$ on the B band are projected onto a reduced 3D representation using the UMAP algorithm.

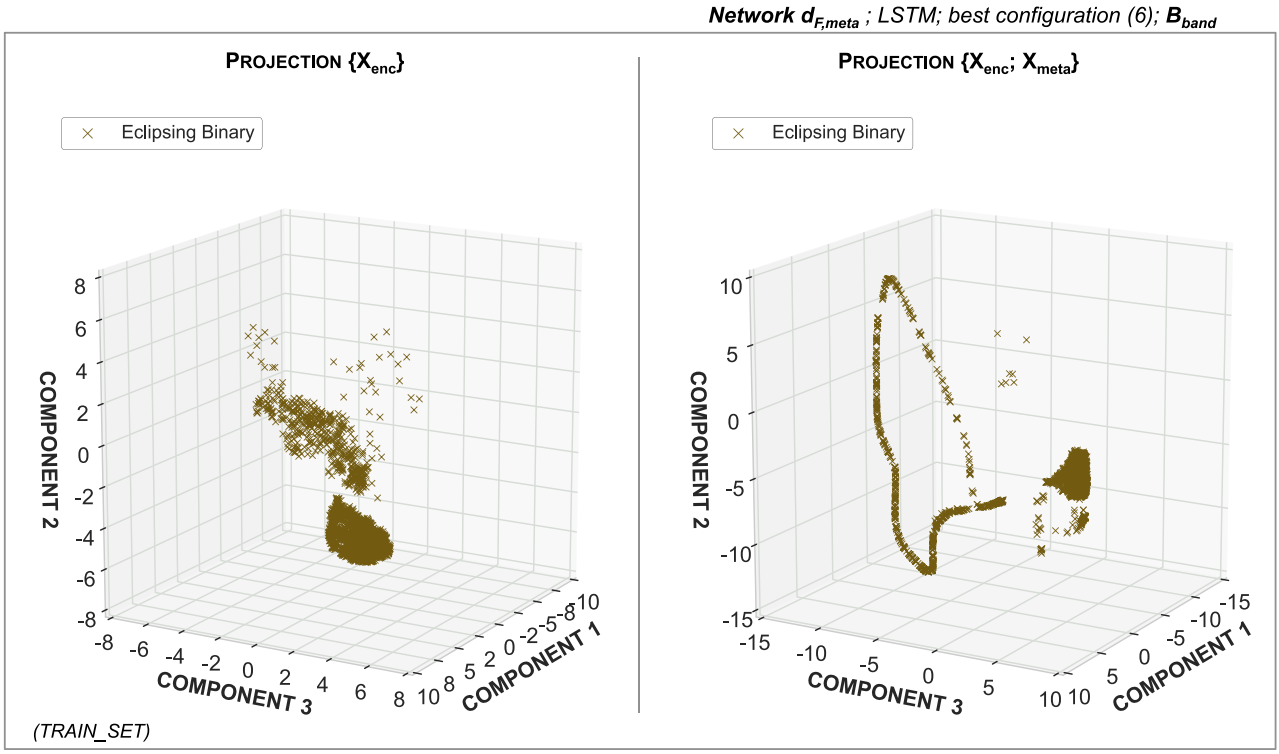

Figure E11. Three-dimensional representation of encoded features for the eclipsing binaries in MACHO. The generated features from the best-performing LSTM composite network $d_{F,\mathrm{meta}}$ on the B band are projected onto a reduced 3D representation using the UMAP algorithm.

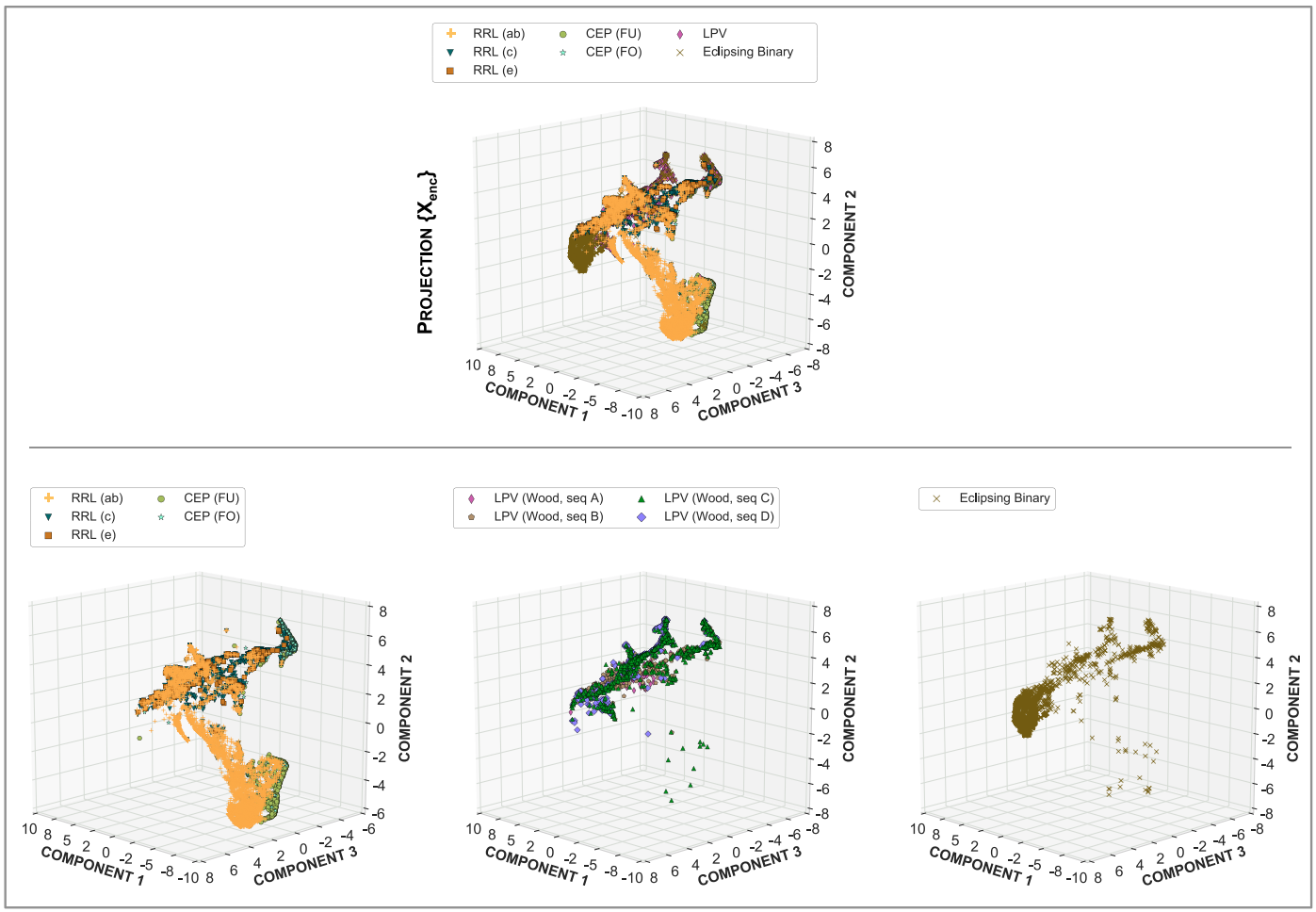

Figure E12. Three-dimensional representation of encoded features for the best-performing LSTM direct classifier $c_{F,\mathrm{meta}}$ on the B band. Generated encodings are projected onto a reduced 3D representation using the UMAP algorithm.

Table E1
Hyperparameter Set Configurations Identified for the Best-performing Networks Based on Minimum Loss Obtained on the Test Set across Three Different Data Sets (*B*-band Only (top), and Two Variants of the Combination of *R* and *B* Bands (Middle and Bottom))

MACHO – $B_{\rm band}$	ID Net	LSTM	GRU	tCNN	dTCN
	c_F	*(6)*	*(6)*	*(3)*	*(1)*
	$c_{F,\rm meta}$	*(1)*	*(5)*	*(3)*	*(1)*
	d_F	*(5)*	*(5)*	*(4)*	*(4)*
	$d_{F,\rm meta}$	*(6)*	*(4)*	*(5)*	*(1)*
MACHO – $RB_{\rm merged}$	ID Net	LSTM	GRU	tCNN	dTCN
	c_F	*(6)*	*(6)*	*(3)*	*(3)*
	$c_{F,\rm meta}$	*(1)*	*(5)*	*(2)*	*(1)*
	d_F	*(5)*	*(5)*	*(5)*	*(5)*
	$d_{F,\rm meta}$	*(6)*	*(4)*	*(6)*	*(4)*
MACHO – $RB_{\rm hybrid}$	ID Net	LSTM	GRU	tCNN	dTCN
	c_F	*(5)*	*(3)*	*(3)*	*(1)*
	$c_{F,\rm meta}$	*(3)*	*(2)*	*(6)*	*(1)*
	d_F	*(5)*	*(5)*	*(5)*	*(4)*
	$d_{F,\rm meta}$	*(5)*	*(1)*	*(4)*	*(4)*

Note. The identifiers *(1)* to *(6)* are stated in Table D2.

Table E2

Classification Accuracy Evaluated on the Test Set for the Best-performing Networks (See the Text for a Description) across Three Different Data Sets (*B*-band Only (Top), and Two Variants of the Combination of *R* and *B* Bands (Middle and Bottom))

MACHO – $B_{\rm band}$	ID Net	LSTM				GRU				tCNN				dTCN			
		Full	Group1	Group2	Group3	Full	Group1	Group2	Group3	Full	Group1	Group2	Group3	Full	Group1	Group2	Group3
	c_F	0.749	0.819	0.802	0.428	0.781	0.834	0.831	0.520	0.732	0.811	0.797	0.359	0.675	0.765	0.752	0.247
	$c_{F,\rm meta}$	0.916	0.936	0.886	0.879	0.907	0.924	0.905	0.850	0.887	0.909	0.890	0.801	0.786	0.807	0.868	0.608
	d_F	0.730	0.798	0.787	0.407	0.739	0.811	0.772	0.431	0.701	0.781	0.760	0.337	0.689	0.782	0.777	0.239
	$d_{F,\rm meta}$	0.905	0.930	0.887	0.833	0.886	0.943	0.907	0.652	0.900	0.929	0.838	0.870	0.802	0.848	0.819	0.611
MACHO – $RB_{\rm merged}$	ID Net	LSTM				GRU				tCNN				dTCN			
		Full	Group1	Group2	Group3	Full	Group1	Group2	Group3	Full	Group1	Group2	Group3	Full	Group1	Group2	Group3
	c_F	0.737	0.808	0.755	0.455	0.780	0.834	0.819	0.532	0.722	0.794	0.785	0.379	0.667	0.729	0.768	0.315
	$c_{F,\rm meta}$	0.890	0.913	0.856	0.845	0.910	0.933	0.888	0.852	0.815	0.848	0.805	0.707	0.747	0.783	0.772	0.582
	d_F	0.726	0.810	0.771	0.364	0.738	0.813	0.779	0.412	0.691	0.770	0.774	0.298	0.686	0.769	0.762	0.290
	$d_{F,\rm meta}$	0.906	0.924	0.896	0.854	0.883	0.938	0.907	0.653	0.912	0.935	0.894	0.848	0.814	0.856	0.864	0.594
MACHO – $RB_{\rm hybrid}$	ID Net	LSTM				GRU				tCNN				dTCN			
		Full	Group1	Group2	Group3	Full	Group1	Group2	Group3	Full	Group1	Group2	Group3	Full	Group1	Group2	Group3
	c_F	0.776	0.841	0.809	0.495	0.789	0.852	0.808	0.535	0.744	0.821	0.779	0.412	0.696	0.768	0.755	0.359
	$c_{F,\rm meta}$	0.917	0.935	0.914	0.857	0.905	0.919	0.892	0.867	0.845	0.858	0.866	0.768	0.768	0.801	0.816	0.588
	d_F	0.748	0.819	0.789	0.434	0.749	0.821	0.772	0.456	0.706	0.788	0.779	0.311	0.726	0.803	0.777	0.379
	$d_{F,\rm meta}$	0.905	0.921	0.894	0.859	0.880	0.939	0.890	0.648	0.904	0.933	0.881	0.825	0.818	0.873	0.872	0.545

Note. The classification accuracy is evaluated for the three main variability groups: short-period pulsators (group 1), eclipsing binaries (group 2), and LPVs (group 3).

Table E3

Classification Metrics Evaluated on the Test Set for the Best-performing Networks (See the Text for a Description) across Three Different Data Sets (*B*-band Only (Top), and Two Variants of the Combination of *R* and *B* Bands (Middle and Bottom))

MACHO – $B_{\rm band}$	ID Net	LSTM			GRU			tCNN			dTCN		
		Precision$_M$	Recall$_M$	F1-score$_M$	Precision$_M$	Recall$_M$	F1-score$_M$	Precision$_M$	Recall$_M$	F1-score$_M$	Precision$_M$	Recall$_M$	F1-score$_M$
	c_F	0.503	0.515	0.504	0.559	0.564	0.552	0.510	0.487	0.488	0.447	0.424	0.425
	$c_{F,\rm meta}$	0.771	0.800	0.784	0.763	0.780	0.770	0.744	0.753	0.748	0.642	0.599	0.609
	d_F	0.477	0.497	0.484	0.500	0.504	0.500	0.459	0.452	0.454	0.446	0.418	0.426
	$d_{F,\rm meta}$	0.765	0.774	0.768	0.679	0.705	0.689	0.748	0.796	0.768	0.597	0.627	0.603
MACHO – $RB_{\rm merged}$	ID Net	LSTM			GRU			tCNN			dTCN		
		Precision$_M$	Recall$_M$	F1-score$_M$	Precision$_M$	Recall$_M$	F1-score$_M$	Precision$_M$	Recall$_M$	F1-score$_M$	Precision$_M$	Recall$_M$	F1-score$_M$
	c_F	0.504	0.519	0.507	0.539	0.567	0.548	0.472	0.483	0.474	0.373	0.389	0.380
	$c_{F,\rm meta}$	0.744	0.757	0.749	0.759	0.794	0.775	0.667	0.673	0.668	0.565	0.566	0.562
	d_F	0.481	0.475	0.477	0.499	0.498	0.496	0.450	0.423	0.433	0.444	0.431	0.435
	$d_{F,\rm meta}$	0.773	0.778	0.774	0.677	0.702	0.687	0.765	0.788	0.776	0.614	0.627	0.619
MACHO – $RB_{\rm hybrid}$	ID Net	LSTM			GRU			tCNN			dTCN		
		Precision$_M$	Recall$_M$	F1-score$_M$	Precision$_M$	Recall$_M$	F1-score$_M$	Precision$_M$	Recall$_M$	F1-score$_M$	Precision$_M$	Recall$_M$	F1-score$_M$
	c_F	0.561	0.559	0.549	0.564	0.570	0.563	0.520	0.520	0.518	0.468	0.480	0.470
	$c_{F,\rm meta}$	0.784	0.790	0.785	0.770	0.785	0.776	0.704	0.691	0.694	0.615	0.600	0.606
	d_F	0.505	0.518	0.508	0.516	0.522	0.516	0.457	0.443	0.447	0.478	0.481	0.478
	$d_{F,\rm meta}$	0.763	0.783	0.772	0.671	0.708	0.688	0.747	0.774	0.758	0.534	0.578	0.551

Table E4
Classification Metrics Computed on the Test Set for the Best-performing LSTM Direct Classifiers c_F and $c_{F,\mathrm{meta}}$ on the B Band

Network c_F; LSTM; *Best Configuration (6);* B_{band}

MACHO – B_{band}	Classes $\{i\}_{i=1\rightarrow N_C}$	NB OF $\boldsymbol{Y}_{\mathrm{true}}$ *(in counts)*	NB OF $\boldsymbol{Y}_{\mathrm{pred}}$ *(in counts)*	TP(i)	FN(i)	FP(i)	TN(i)	Precision(i)	Recall(i)	F1-score(i)
	RR Lyrae (type *ab*)	1430	1574	1370	60	204	1890	0.870	0.958	0.912
	RR Lyrae (type *c*)	344	333	172	172	161	3019	0.517	0.500	0.508
	RR Lyrae (type *e*)	60	0	0	60	0	3464	…	…	…
	CEP (FU)	229	202	174	55	28	3267	0.861	0.760	0.807
	CEP (FO)	133	153	82	51	71	3320	0.536	0.617	0.573
	LPV (Wood seq. A)	62	0	0	62	0	3462	…	…	…
	LPV (Wood seq. B)	160	289	87	73	202	3162	0.301	0.544	0.388
	LPV (Wood seq. C)	220	164	64	156	100	3204	0.390	0.291	0.333
	LPV (Wood seq. D)	152	158	103	49	55	3317	0.652	0.678	0.665
	Eclipsing binaries	734	651	589	145	62	2728	0.905	0.802	0.851
	$N_C = 10$	$N_s = 3524$					Accuracy	Precision$_M$	Recall$_M$	F1-score$_M$
							0.749	0.503	0.515	0.504

Network $c_{F,\mathrm{meta}}$; LSTM; *Best Configuration (1);* B_{band}

MACHO – B_{band}	Classes $\{i\}_{i=1\rightarrow N_C}$	NB OF $\boldsymbol{Y}_{\mathrm{true}}$ *(in counts)*	NB OF $\boldsymbol{Y}_{\mathrm{pred}}$ *(in counts)*	TP(i)	FN(i)	FP(i)	TN(i)	Precision(i)	Recall(i)	F1-score(i)
	RR Lyrae (type *ab*)	1430	1460	1422	8	38	2056	0.974	0.994	0.984
	RR Lyrae (type *c*)	344	388	316	28	72	3108	0.814	0.919	0.863
	RR Lyrae (type *e*)	60	0	0	60	0	3464	…	…	…
	CEP (FU)	229	232	204	25	28	3267	0.879	0.891	0.885
	CEP (FO)	133	141	113	20	28	3363	0.801	0.850	0.825
	LPV (Wood seq.A)	62	65	49	13	16	3446	0.754	0.790	0.772
	LPV (Wood seq. B)	160	157	131	29	26	3338	0.834	0.819	0.826
	LPV (Wood seq. C)	220	230	195	25	35	3269	0.848	0.886	0.867
	LPV (Wood seq. D)	152	175	147	5	28	3344	0.840	0.967	0.899
	Eclipsing binaries	734	676	650	84	26	2764	0.962	0.886	0.922
	$N_C = 10$	$N_s = 3524$					Accuracy	Precision$_M$	Recall$_M$	F1-score$_M$
							0.916	0.771	0.800	0.784

Table E5
Classification Metrics Computed on the Test Set Using the Metadata via the Classifier Module

Network Classifier Module (MLP); Metadata

MACHO—Metadata	Classes $\{i\}_{i=1\rightarrow N_C}$	Nb of $\boldsymbol{Y}_{\mathrm{true}}$ *(in counts)*	Nb of $\boldsymbol{Y}_{\mathrm{pred}}$ *(in counts)*	TP(i)	FN(i)	FP(i)	TN(i)	Precision(i)	Recall(i)	F1-score(i)
	RR Lyrae (type *ab*)	1430	1457	1414	16	43	2051	0.970	0.989	0.980
	RR Lyrae (type *c*)	344	397	323	21	74	3106	0.814	0.939	0.872
	RR Lyrae (type *e*)	60	0	0	60	0	3464	…	…	…
	CEP (FU)	229	254	213	16	41	3254	0.839	0.930	0.882
	CEP (FO)	133	114	95	38	19	3372	0.833	0.714	0.769
	LPV (Wood seq. A)	62	50	42	20	8	3454	0.840	0.677	0.750
	LPV (Wood seq. B)	160	265	158	2	107	3257	0.596	0.988	0.744
	LPV (Wood seq. C)	220	0	0	220	0	3304	…	…	…
	LPV (Wood seq. D)	152	0	0	152	0	3372	…	…	…
	Eclipsing binaries	734	987	686	48	301	2489	0.695	0.935	0.797
	$N_C = 10$	$N_s = 3524$					Accuracy	Precision$_M$	Recall$_M$	F1-score$_M$
							0.832	0.559	0.617	0.579

ORCID iDs

Sara Jamal https://orcid.org/0000-0002-3929-6668
Joshua S. Bloom https://orcid.org/0000-0002-7777-216X

References

Abadi, M., Agarwal, A., & Barham, P. 2016a, arXiv:1603.04467
Abadi, M., Barham, P., Chen, J., et al. 2016b, in Proc. 12th USENIX Symp. on Operating Systems Design and Implementation (Savannah, GA: USENIX), 265
Alcock, C., Allsman, R. A., Axelrod, T. S., et al. 1996, ApJ, 461, 84
Alves, D. R. 2004, NewAR, 48, 659
Ambikasaran, S., Foreman-Mackey, D., Greengard, L., Hogg, D. W., & O'Neil, M. 2016, ITPAM, 38, 252
Aniyan, A. K., & Thorat, K. 2017, ApJS, 230, 20
Ansdell, M., Ioannou, Y., Osborn, H. P., et al. 2018, ApJL, 869, L7
Armstrong, D. J., Kirk, J., Lam, K. W. F., et al. 2016, MNRAS, 456, 2260
Barbary, K., Biswas, K., Goldstein, R., et al. 2016, sncosmo/sncosmo v1.4.0, Zenodo, doi:10.5281/zenodo.168220
Blomme, J., Sarro, L. M., O'Donovan, F. T., et al. 2011, MNRAS, 418, 96
Boone, K. 2019, ApJ, 158, 257
Breiman, L. 2001, Machine Learning, 45, 5
Brunel, A., Pasquet, J., Pasquet, J., et al. 2019, Electronic Imaging, 2019, 90
Cabral, J. B., Sánchez, B., Ramos, F., et al. 2018, A&C, 25, 213
Cabrera-Vives, G., Reyes, I., Förster, F., Estévez, P. A., & Maureira, J.-C. 2016, in Proc. Int. Joint Conf. on Neural Networks, ed. P. A. Estévez (Pisataway, NJ: IEEE), 251, doi:10.1109/IJCNN.2016.7727206
Carretta, E., Gratton, R. G., Clementini, G., & Pecci, F. F. 2000, ApJ, 533, 215
Charnock, T., & Moss, A. 2017, ApJL, 837, L28
Cho, K., van Merrienboer, B., Gulcehre, C., et al. 2014, arXiv:1406.1078
Clarke, A., Scaife, A., Greenhalgh, R., & Griguta, V. 2020, A&A, 639, 84
Clementini, G., Gratton, R., Bragaglia, A., et al. 2003, AJ, 125, 1309
Connor, L., & Leeuwen, J. v. 2018, AJ, 156, 256
Cook, K. H., Alcock, C., Allsman, R. A., et al. 1995, IAUCo, 155, 221
D'Isanto, A., Cavuoti, S., Brescia, M., et al. 2016, MNRAS, 457, 3119
D'Isanto, A., & Polsterer, K. L. 2018, A&A, 609, A111
Debosscher, J., Sarro, L. M., Aerts, C., et al. 2007, A&A, 475, 1159
Dékány, I., Hajdu, G., Grebel, E. K., & Catelan, M. 2019, ApJ, 883, 58
Derekas, A., Kiss, L. L., & Bedding, T. R. 2007, ApJ, 663, 249
Dieleman, S., Willett, K. W., & Dambre, J. 2015, MNRAS, 450, 1441
Domínguez Sánchez, H., Huertas-Company, M., Bernardi, M., Tuccillo, D., & Fischer, J. L. 2018, MNRAS, 476, 3661
Dubath, P., Rimoldini, L., Süveges, M., et al. 2011, MNRAS, 414, 2602
Erdmann, M., Glombitza, J., & Walz, D. 2018, APh, 97, 46
Eyer, L., & Mowlavi, N. 2008, J. Phys. Conf. Ser., 118, 012010
Fan, X., Li, J., Li, X., Zhong, Y., & Cao, J. 2019, SCPMA, 62, 969512
Foreman-Mackey, D. 2018, RNAAS, 2, 31
Foreman-Mackey, D., Agol, E., Ambikasaran, S., & Angus, R. 2017, AJ, 154, 220
Foreman-Mackey, D., Barentsen, G., & Barclay, T. 2019, dfm/exoplanet: exoplanet v0.1.6, Zenodo, doi:10.5281/zenodo.2651251
Freedman, W. L., Madore, B. F., Gibson, B. K., et al. 2001, ApJ, 553, 47
Friedman, J. H. 1984, A Variable Span Smoother Tech. Rep. LCS-TR-5, (Stanford, CA: Stanford Univ. CA Lab for Computational Statistics)
Gabbard, H., Williams, M., Hayes, F., & Messenger, C. 2018, PhRvL, 120, 141103
Gebhard, T., Kilbertus, N., Parascandolo, G., Harry, I., & Schölkopf, B. 2017, in Proc. 31st Conf. on Neural Information Processing Systems (San Diego, CA: NIPS)
Gebhard, T. D., Kilbertus, N., Harry, I., & Schölkopf, B. 2019, PhRvD, 100, 063015
George, D., & Huerta, E. A. 2018a, PhLB, 778, 64
George, D., & Huerta, E. A. 2018b, PhRvD, 97, 044039
Gillet, N., Mesinger, A., Greig, B., Liu, A., & Ucci, G. 2019, MNRAS, 484, 282
Hartman, J. D., & Bakos, G. A. 2016, A&C, 17, 1
Hochreiter, S., & Schmidhuber, J. 1997, Neural Comput., 9, 1735
Hon, M., Stello, D., & Yu, J. 2017, MNRAS, 469, 4578
Hoyle, B. 2016, A&C, 16, 34
Huang, C. D., Riess, A. G., Hoffmann, S. L., et al. 2018, ApJ, 857, 67
Ishida, E. E. O. 2019, NatAs, 3, 680
Ivezić, v., Kahn, S. M., Tyson, J. A., et al. 2019, ApJ, 873, 111
Jacobs, C., Collett, T., Glazebrook, K., et al. 2019, MNRAS, 484, 5330
Johnston, K. B., Caballero-Nieves, S. M., Petit, V., Peter, A. M., & Haber, R. 2020, MNRAS, 491, 3805
Kim, D.-W., & Bailer-Jones, C. A. L. 2016, A&A, 587, A18
Kim, E. J., & Brunner, R. J. 2017, MNRAS, 464, 4463
Kingma, D. P., & Ba, J. 2017, arXiv:1412.6980
Kraft, R. P., & Schmidt, M. 1963, ApJ, 137, 249
Lanusse, F., Ma, Q., Li, N., et al. 2018, MNRAS, 473, 3895
Laureijs, R., Amiaux, J., Arduini, S., et al. 2011, arXiv:1110.3193
Lea, C., Flynn, M. D., Vidal, R., Reiter, A., & Hager, G. D. 2017, in Proc. IEEE Conf. on Computer Vision and Pattern Recognition (Piscataway, NJ: IEEE), 156
Lomb, N. R. 1976, Ap&SS, 39, 447
Majaess, D. J., Turner, D. G., & Lane, D. J. 2009, MNRAS, 398, 263
Masci, F. J., Hoffman, D. I., Grillmair, C. J., & Cutri, R. M. 2014, AJ, 148, 21
McInnes, L., Healy, J., & Melville, J. 2018, arXiv:1802.03426
Möller, A., & de Boissière, T. 2019, MNRAS, 491, 4277
Mustafa, M., Bard, D., Bhimji, W., et al. 2019, ComAC, 6, 1
Muthukrishna, D., Narayan, G., Mandel, K. S., Biswas, R., & HloŽek, R. 2019a, PASP, 131, 118002
Muthukrishna, D., Parkinson, D., & Tucker, B. E. 2019b, ApJ, 885, 85
Naul, B., Bloom, J. S., Pérez, F., & van der Walt, S. 2017, A Recurrent Neural Network for Classification of Unevenly Sampled Variable Stars v1.0.0, Zenodo, doi:10.5281/zenodo.1045560
Naul, B., Bloom, J. S., Pérez, F., & van der Walt, S. 2018, NatAs, 2, 151
Naul, B., van der Walt, S., Crellin-Quick, A., Bloom, J., & Pérez, F. 2016, in Proc. 15th Python in Science Conf., ed. S. Benthall & S. Rostrup (Austin, TX: SciPy), 27, doi:10.25080/Majora-629e541a-004
Ntampaka, M., Eisenstein, D. J., Yuan, S., & Garrison, L. H. 2020, ApJ, 889, 151
Nun, I., Protopapas, P., Sim, B., et al. 2015, arXiv:1506.00010v2
Oliphant, T. E. 2006, A Guide to NumPy
Oord, A. v. d., Dieleman, S., Zen, H., et al. 2016, arXiv:1609.03499
Paczyński, B. 1997, in Proc. STScI May Symp., ed. M. Livio, M. Donahue, & N. Panagia (Cambridge: Cambridge Univ. Press), 273
Parks, D., Prochaska, J. X., Dong, S., & Cai, Z. 2018, MNRAS, 476, 1151
Pasquet, J., Bertin, E., Treyer, M., Arnouts, S., & Fouchez, D. 2019a, A&A, 621, A26
Pasquet, J., Pasquet, J., Chaumont, M., & Fouchez, D. 2019b, A&A, 627, A21
Pedregosa, F., Varoquaux, G., Gramfort, A., et al. 2011, J. Machine Learning Research, 12, 2825
Pieringer, C., Pichara, K., Catelán, M., & Protopapas, P. 2019, MNRAS, 484, 3071
Pruzhinskaya, M. V., Malanchev, K. L., Kornilov, M. V., et al. 2019, MNRAS, 489, 3591
Reis, I., Rotman, M., Poznanski, D., Prochaska, J. X., & Wolf, L. 2019, arXiv:1911.06823
Richards, J. W., Starr, D. L., Butler, N. R., et al. 2011, ApJ, 733, 10
Richards, J. W., Starr, D. L., Miller, A. A., et al. 2012, ApJS, 203, 32
Riess, A. G., Casertano, S., Yuan, W., et al. 2018, ApJ, 861, 126
Riess, A. G., Casertano, S., Yuan, W., Macri, L. M., & Scolnic, D. 2019, ApJ, 876, 85
Rimoldini, L., Dubath, P., Süveges, M., et al. 2012, MNRAS, 427, 2917
Salvatier, J., Wiecki, T. V., & Fonnesbeck, C. 2016, PeerJ Computer Science, 2, e55
Samus', N. N., Kazarovets, E. V., Durlevich, O. V., Kireeva, N. N., & Pastukhova, E. N. 2017, ARep, 61, 80
Scargle, J. D. 1998, ApJ, 504, 405
Schanche, N., Cameron, A. C., Hébrard, G., et al. 2019, MNRAS, 483, 5534
Schmidhuber, J. 2015, NN, 61, 85
Shallue, C. J., & Vanderburg, A. 2018, AJ, 155, 94
Shen, H., George, D., Huerta, E. A., & Zhao, Z. 2019, in Proc. IEEE Int. Conf. on Acoustics, Speech and Signal Processing (ICASSP) (Piscataway, NJ: IEEE), 3237, doi:10.1109/ICASSP.2019.8683061
Skowron, D. M., Skowron, J., Mróz, P., et al. 2019, Sci, 365, 478
Spergel, D., Gehrels, N., Baltay, C., et al. 2015, arXiv:1503.03757
Srivastava, N., Hinton, G., Krizhevsky, A., Sutskever, I., & Salakhutdinov, R. 2014, J. Machine Learning Research, 15, 1929
Stetson, P. B. 1996, PASP, 108, 851
Theano Development Team, Al-Rfou, R., Alain, G., et al. 2016, arXiv:1605.02688

Torres, G., Andersen, J., & Giménez, A. 2010, A&ARv, 18, 67
Tsang, B. T.-H., & Schultz, W. C. 2019, ApJL, 877, L14
Vanderplas, J., Naul, B., Willmer, A., Williams, P., & Morris, B. M. 2016, gatspy: Feature Release v0.3, Zenodo, doi:10.5281/zenodo.47887
van der Walt, S. v. d., Colbert, S. C., & Varoquaux, G. 2011, CSE, 13, 22
VanderPlas, J. T., & Ivezić, v. 2015, ApJ, 812, 18
Wood, P., Olivier, E., & Kawaler, S. 2004, ApJ, 604, 800
Wood, P. R., Alcock, C., Allsman, R. A., et al. 1999, in IAU Symp. 191, Asymptotic Giant Branch Stars, ed. T. Le Bertre, A. Lebre, & C. Waelkens (Cambridge: Cambridge Univ. Press), 151
Zong, B., Song, Q., Min, M. R., et al. 2018, in Proc. 6th Int. Conf. on Learning Representations, ed. Y. Bengio et al. (La Jolla, CA: ICRL), #27