Monthly Notices
of the
ROYAL ASTRONOMICAL SOCIETY

MNRAS **491,** 4277–4293 (2020)
Advance Access publication 2019 December 3

doi:10.1093/mnras/stz3312

SuperNNova: an open-source framework for Bayesian, neural network-based supernova classification

A. Möller [1,2,3]★ and T. de Boissière[4]†

[1]*Research School of Astronomy and Astrophysics, Australian National University, Canberra, ACT 2611, Australia*
[2]*ARC Centre of Excellence for All-sky Astrophysics (CAASTRO)*
[3]*Université Clermont Auvergne, CNRS/IN2P3, LPC, F-63000 Clermont-Ferrand, France*
[4]*Lyrebird AI, Unit 302, 55 Mont-Royal Avenue Ouest, Montreal, QC H2T 2S5, Canada*

Accepted 2019 November 23. Received 2019 November 23; in original form 2019 January 17

ABSTRACT

We introduce SuperNNova, an open-source supernova photometric classification framework that leverages recent advances in deep neural networks. Our core algorithm is a recurrent neural network (RNN) that is trained to classify light curves using only photometric information. Additional information such as host-galaxy redshift can be incorporated to improve performance. We evaluate our framework using realistic supernova simulations that include survey detection. We show that our method, for the type Ia versus non-Ia supernova classification problem, reaches accuracies greater than 96.92 ± 0.09 without any redshift information and up to 99.55 ± 0.06 when redshift, either photometric or spectroscopic, is available. Further, we show that our method attains unprecedented performance for the classification of incomplete light curves, reaching accuracies >86.4 ± 0.1 (>93.5 ± 0.8) without host-galaxy redshift (with redshift information) 2 d before maximum light. In contrast with previous methods, there is no need for time-consuming feature engineering and we show that our method scales to very large data sets with a modest computing budget. In addition, we investigate often neglected pitfalls of machine learning algorithms. We show that commonly used algorithms suffer from poor calibration and overconfidence on out-of-distribution samples when applied to supernova data. We devise extensive tests to estimate the robustness of classifiers and cast the learning procedure under a Bayesian light, demonstrating a much better handling of uncertainties. We study the benefits of Bayesian RNNs for SN Ia cosmology. Our code is open sourced and available on github[1].

Key words: methods: data analysis – methods: observational – supernovae: general – cosmology: observational.

1 INTRODUCTION

To study the expansion of the Universe, increasingly large samples of Type Ia supernovae (SNe Ia) are used to measure luminosity distances as a function of redshift. Traditionally, SNe Ia are classified after a spectroscopic follow-up of promising supernova candidates identified by photometric surveys. Previous and current surveys such as SNLS, SDSS, the Foundation Supernova Survey, and the Dark Energy Survey have discovered thousands of supernovae (Astier et al. 2006; Frieman et al. 2008; Bernstein et al. 2012; Foley et al. 2018). Of those supernovae, only a small number can be spectroscopically followed up. The upcoming Large Synoptic Survey Telescope (LSST) survey is expected to discover 10^7 supernovae (LSST Science Collaboration 2009). This survey, together with others such as the Zwicky Transient Survey, will explore transient events and expect to have up to a million transient alerts per night (Bellm 2019). The scarcity of spectroscopic resources is, and will continue to be, a limiting factor for supernova cosmology and time-domain astronomy. Thus, in order to fully take advantage of the available data, fast and accurate photometric classification is crucial.

In recent years, different methods have been developed to classify supernovae using their light curves, i.e. their observed brightness evolution over time in different colour bands. This is a challenging problem since brightness measurements are often noisy and non-uniform in time. Most classification methods rely either on light-curve template matching (Poznanski, Maoz & Gal-Yam 2007; Sako et al. 2011) or classification using features extracted from

★ E-mail: anais.moller@clermont.in2p3.fr
† Equal contribution.

template or functional fits. Extracted features can be used to classify supernovae using sequential cuts (Bazin et al. 2011; Campbell et al. 2013) or machine learning algorithms (Kessler et al. 2010b; Lochner et al. 2016; Möller et al. 2016; Dai et al. 2018). Recent advances in deep learning, a branch of machine learning, have shown that neural network classifiers trained on raw data can outperform classifiers based on handcrafted features (Charnock & Moss 2017; Kimura et al. 2017; Moss 2018). In this work, we explore this promising direction and obtain state-of-the-art results on a variety of supernova classification tasks.

Typically, the performance of classification algorithms is assessed on simulated data. In particular, many algorithms have used the 'Supernova Photometric Classification Challenge' (SPCC; Kessler et al. 2010a) data set for training and evaluation. Some of these methods have also been successfully applied to real survey data, such as SDSS (Sako et al. 2011) and SNLS (Kuznetsova & Connolly 2007; Poznanski et al. 2007; Bazin et al. 2011; Möller et al. 2016). Recently, in preparation for LSST, an open data challenge called 'Photometric LSST Astronomical Time Series Classification Challenge (PLAsTiCC)' (The PLAsTiCC team et al. 2018) was launched in an effort to better classify simulated astronomical time-series data.

Existing classification algorithms have been shown to obtain high-purity SN Ia samples with complete light-curve information. However, only a handful are able to classify SNe within a limited number of photometric epochs (Poznanski et al. 2007; Sako et al. 2011; Ishida & de Souza 2013; Charnock & Moss 2017; Kimura et al. 2017). Partial light-curve classification is key to prioritizing spectroscopic and photometric resources for promising candidates.

Photometrically classified supernovae can be used in statistical analyses such as rates and SN Ia cosmology (González-Gaitán et al. 2011; Campbell et al. 2013; Jones et al. 2018) but those analyses suffer from bias induced by contamination from other SN types. For type Ia SN cosmology, this bias has been accounted for using Bayesian frameworks (Hlozek et al. 2012; Hinton et al. 2018; Jones et al. 2018) but little attention has been devoted to the calibration of the underlying machine learning classifiers. This should be an important concern given that prior work has demonstrated that miscalibration was a common issue in machine learning (Niculescu-Mizil & Caruana 2005; Guo et al. 2017). Another pitfall is that commonly used algorithms give no information about the uncertainty of their predictions. Recent advances in Bayesian neural networks (Gal & Ghahramani 2015a, b; Fortunato, Blundell & Vinyals 2017) provide a framework for training neural networks that are able to estimate model uncertainty in a computationally efficient way.

Model uncertainties can provide valuable information on the completeness of the training set. It has been shown that the data samples commonly used to train supernova classifiers are far from complete (Lochner et al. 2016; Ishida et al. 2018). These samples are further biased by the fact that they only contain known supernova types that match specific templates. New classes of transients may be found in the survey sample, and existing classification algorithms will still try to assign it to one of the known classes or targets. Such events, so-called out-of-distribution (OOD) events, can contaminate a photometrically selected sample. To avoid this, some analyses advocated for pre-selection cuts (Kuznetsova & Connolly 2007; Möller et al. 2016) to restrict candidates to a subset of known classes. However, this prevents the analysis of new cosmological objects, a task that will be a major focus of brokers such as ANTARES (Narayan et al. 2018).

In this work, we present SuperNNova, an open-source algorithm that fits the requirements highlighted above and alleviates some of the concerns linked to the use of machine learning. SuperNNova is based on raw, photometric time-series inputs. Thus, it is neither biased by template matching nor limited by costly feature engineering.

SuperNNova is the first SN light-curve classification framework that uses Bayesian neural networks. For the first time in astronomical analyses, two variational inference methods (Gal & Ghahramani 2015a; Fortunato et al. 2017) are available in the same framework and compared using SN simulations. Furthermore, we are able to handle irregular time-series without the use of data augmentation or fitting methods. This is crucial for telescope observations that are subject to weather condition changes and non-uniform scheduling.

With SuperNNova, we obtain state-of-the-art results on the classification of supernovae with complete and partial light-curve data. We show that the SuperNNova's performance increases with the amount of data and show that it readily scales to very large data sets. The performance is further enhanced with additional information such as host-galaxy redshifts. Crucially, we show that the SuperNNova's classifiers can be trained in a principled, Bayesian way and yield calibrated predictions with sensible uncertainty estimates. Those uncertainty estimates make SuperNNova better suited to the task of identifying OOD candidates as we show it is not susceptible to classifying those samples with high confidence.

Although this method has been designed with supernova cosmology in mind, SuperNNova can also be applied to any light-curve classification task in time-domain astronomy.

This manuscript is structured as follows. We will introduce the simulations, metrics, and machine learning algorithms used in Section 2. Then we will present and study the performance of the three main algorithms available in SuperNNova: a baseline RNN (Section 3) and two Bayesian RNNs (Section 4). In Section 6, we briefly outline the computational resources required to use SuperNNova. In Section 5, we devise extensive tests to evaluate the statistical robustness of our classifiers and analyse the effect of a photometrically classified type Ia SN sample in cosmology.

2 METHODS

2.1 Simulations and available data sets

We simulated realistic supernova light curves using the SNANA package (Kessler et al. 2009). Our data are similar to the SPCC data (Kessler et al. 2010a,b) with updated models used in the Dark Energy Survey (DES) simulations (Kessler et al. 2018). The DES light curves are built from supernova templates and use the SALT2 SN Ia SED models (Guy et al. 2007) and the trained model from Joint Lightcurve Analysis (JLA, Betoule et al. 2014). Observing logs specifying the simulated cadence and conditions were included when available. Our simulation, which is publically available,[2] contains 1983 213 supernova light curves of which 881 969 are successfully fitted using SALT2 to obtain features such as colour and stretch. SALT2 fits do not always converge and therefore some light curves cannot be fitted. Furthermore, we require loose cuts for fitting, such as: an observation with S/N $>$ 5, and at least 1 epoch before -2 d and at least 1 epoch past $+30$ d (rest frame).

[2]DOI:10.5281/zenodo.3265189

Table 1. Simulated supernova samples divided by subtypes. We show the number of supernova light curves available per type in the complete data set and the number that has a successful SALT2 fit. Each SN type represents a template with the exception of type IIL for which we use two different templates.

Simulated supernovae		
SN type	SALT2 fitted	Complete data set
Ia	402 786	912 691
Ib	140 197	181 454
Ic	70 811	90 485
IIP	94 994	296 523
IIn	3249	154 614
IIL	93 535	189 615
	IILs by template	
IIL1	26 717	100 827
IIL2	66 818	88 788

The simulation includes spectroscopic and photometric host-galaxy redshifts (Kessler et al. 2010a; Gupta et al. 2016). To reflect real survey conditions, spectroscopic redshift is obtained only for a subset of light curves. In all the classification tasks that follow, we subsample the data set to make sure classes are balanced. For binary classification, i.e. the discrimination of SNe Ia versus others, we obtain a balanced sample of 912 691 light curves (resp. 402 786 with SALT2 fits). Detailed statistics for binary classification are shown in Table 1. Our simulation count is more than an order of magnitude larger than SPCC, which contained 21 319 light curves. We split the data as follows: 80 per cent training, 10 per cent validation, and 10 per cent testing. In the following, we report our results separately for the *'complete'* and the *'SALT2-fitted'* data sets. The former contains all our simulated light curves after class balancing. The latter contains only light curves that were successfully fitted with SALT2 (Guy et al. 2007). Since past and current supernova surveys have been focused on discovering type Ia supernovae, this *'SALT2-fitted'* sample is closer to their spectroscopically targeted supernovae. We will call the *'SALT2-fitted'* data set a non-representative one, as it fails to explore the full diversity of supernovae. Our representative set will be the *'complete'* data set. We further explore these samples and the issue of representativeness in Section 5.2.

2.2 Evaluation metrics

2.2.1 *ROC curve*

AUC is a robust metric, commonly used as an evaluation method for binary classification. AUC stands for Area Under Curve, where the curve is the ROC curve (Receiver Operating Characteristic). The ROC curve gives an indication of the performance of a binary classifier by plotting the true positive rate (efficiency) against the false positive rate (contamination). While the ROC curve represents the performance of a model in two dimensions, the AUC simplifies this into a number. A perfect model would score an AUC of 1 while a random classifier would score 0.5.

2.2.2 *Accuracy, purity, and contamination*

If classes are balanced, classification accuracy is an intuitive and useful metric. Accuracy is measured as the number of correct predictions against the total number of predictions. For binary classification, accuracy can also be calculated in terms of positives and negatives as follows:

$$\text{Accuracy} = \frac{\text{TP} + \text{TN}}{\text{TP} + \text{TN} + \text{FP} + \text{FN}}, \quad (1)$$

where TP (resp. TN) are true positives (resp. negatives) and FP (resp. FN) are false positives (resp. negatives). For binary classification, TP shows the number of correctly classified SNe Ia while TN shows the number of correctly classified core-collapse SNe. In this work, unless specified, the predicted type corresponds to the highest probability class.

The purity of the SN Ia sample and the classification efficiency are defined as

$$\text{Purity} = \frac{\text{TP}}{\text{TP} + \text{FP}}; \quad \text{Efficiency} = \frac{\text{TP}}{\text{TP} + \text{FN}}. \quad (2)$$

Contamination by core collapse is defined as

$$\text{Contamination} = \frac{\text{FP}}{\text{FP} + \text{TN}}. \quad (3)$$

In this work, performance metrics reported with errors represent the mean and one standard deviation of the distribution. This distribution is obtained by performing five runs with different random seeds.

2.3 Recurrent neural networks (RNNs)

RNNs are a class of neural nets that allow connections between hidden units with a time delay; they are well suited for modelling sequential data or time-series data (Bahdanau, Cho & Bengio 2014; Sutskever, Vinyals & Le 2014; Mehri et al. 2016; Kalchbrenner et al. 2018; Vasquez & Lewis 2019). Through the connections between hidden units, the model can learn to retain or discard past information about its inputs, and in principle discover correlations across time. Popular recent architectures such as long short-term memory (LSTM) (Hochreiter & Schmidhuber 1997) or Gated Recurrent Univt (GRU) (Chung et al. 2014) alleviate some of the issues that occur when training these models. They introduce a memory cell and gating mechanisms that endow recurrent networks with better control of the information flow.

The structure of a basic RNN with two hidden layers is illustrated in Fig. 1. The same series of operations (typically an affine matrix multiplication followed by an activation function) are applied to each element of a sequence $(X)_{t\in[1,T]}$ to produce a sequence of hidden states (h_t). These operations are parametrized by learnt weights (W, W', V, U, and U' in Fig. 1) and biases (b). Multiple recurrent layers can be stacked on top of each other by feeding a layer's output as input to the next layer. To obtain the class prediction, the sequential information is eventually condensed to a fixed-length representation (for instance, through mean pooling, i.e. averaging of the hidden states). The use of a softmax function ensures that the network's output can be understood as probabilities. In Fig. 1, the RNN is unidirectional: It is allowed to process information from left to right, and bidirectional networks are also allowed to process from right to left. This has been used in many applications such as language translation to improve performance.

Traditional RNNs lack the ability to quantify model uncertainty. However, quantifying the uncertainty of a model's prediction is a highly desirable feature for cosmological applications. In this work, we leverage two recent advances in variational inference that have proven effective in helping neural networks to estimate uncertainty. The work of (Gal & Ghahramani 2015b) show that dropout (a regularization technique consisting in randomly dropping

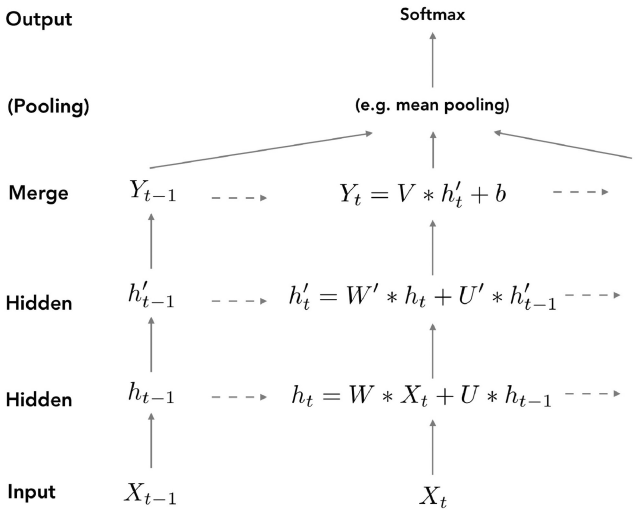

Figure 1. Partial schema of a vanilla RNN with mean pooling. Layers are indicated in the first left column. Inputs (X_t) are given to the network. The arrows indicate transmission of temporal information. The network extends to the right by the number of $t \in [1, T]$ available inputs.

weights or activations) has a Bayesian interpretation and can be used to improve performance and calibrate predictions. Fortunato et al. (2017) optimize a posterior distribution over each of the neural network's weights and have shown competitive results and high-quality uncertainties on challenging language modelling and image captioning tasks. We introduce variational inference and our approximation methods in Section 4.

2.4 RNN architecture

Our PyTorch (Paszke et al. 2017) implementation is publicly available on GitHub[3] with an accompanying documentation.

The core building block of our algorithm is an RNN. Our RNN processes the photometric time series to produce a sequence of hidden states. The sequence is condensed to a fixed length through mean pooling or alternatively by concatenating the last hidden state of each recurrent layer. In the following, we will call these RNN output options mean pooling or standard, respectively. Finally, a linear projection layer is applied to obtain a N-dimensional vector, where N is the number of classes. A softmax function is used to obtain probabilities for an input to belong to a given class.

We make it easy for the user to experiment with hyper-parameters such as layer type, network size, bidirectionality, batch size, or gradient descent optimizer through an extensive command line interface. In addition, we implement two Bayesian training methods: the first one dubbed 'MC Dropout' implements (Gal & Ghahramani 2015b) and the second one that we simply call 'BBB' implements (Fortunato et al. 2017).[4] These two Bayesian methods provide principled regularization for the training of neural networks, and have been shown to provide superior uncertainty estimation, a crucial property for astronomy and cosmology applications. This framework uses the python libraries: pandas, numpy, astropy, matplotlib, pytorch, scikit-learn, and seaborn (Hunter 2007; McKinney 2010; Pedregosa et al. 2011; van der Walt, Colbert & Varoquaux 2011; Astropy Collaboration 2013; Paszke et al. 2017; The Astropy Collaboration 2018; Waskom et al. 2018).

2.4.1 *RNN training data*

The only inputs required by our algorithm are photometric time series or light curves. However, any other information available can and should be used to improve the performance. Thus, we made it straightforward to add additional inputs, such as host-galaxy redshift by simply repeating the redshift value and its error for all time-steps in the time series. There is no need for tedious, time-consuming feature engineering or domain-expert knowledge. This is a decisive advantage of our method since it has been shown (Lochner et al. 2016) that the choice of the feature extraction method can significantly impact classification results.

These data can be used to train RNNs for three distinct classification tasks: binary (SNe Ia versus non-Ia), ternary (Ia, II, and Ibc), and seven-way classification (Ia, IIP, IIn, IIL1, IIL2, Ib, and Ic).

2.4.2 *RNN training details*

The light-curve data are typically formatted in a tabular way, where each row corresponds to one observation, in one specific filter. We group observations within 8 h of each other on a single row, which is equivalent to grouping all the observed fluxes in a given night. If a given filter is not observed, we assign it a special value to indicate that it is missing. We add a delta time feature to indicate how much time has elapsed since the last observation. We also add a feature indicating which filter is available. The features used in the algorithm for a given time-step are: fluxes (FLUXCAL in filters *g*, *i*, *r*, and *z*), flux errors (FLUXCALERR in filters *g*, *i*, *r*, and *z*), time-step (delta_time), and one-hot encoding representing available flux information (*g*, *gi*, *gir*, *girz*, *giz*, *gr*, *grz*, *gz*, *i*, *ir*, *irz*, *iz*, *r*, *rz*, and *z*). Additional information can be provided such as host-galaxy redshift, spectroscopic or photometric, and its error.

Light-curve fluxes and errors exhibit large variations, which is why we paid special care to input data normalization. Three options are supported: no normalization, individual, 'per-filter' normalization, and 'global' normalization. The latter two are obtained as follows: features (f) are first log transformed and then scaled. The log transform (f_l) uses the minimum value of the feature min(f) and a constant (ϵ) to centre the distribution in zero as follows: $f_l = \log[-\min(f) + f + \epsilon]$. Using the mean and standard deviation of the log transform $[\mu, \sigma(f_l)]$, standard scaling is applied: $\hat{f} = [f_l - \mu(f_l)]/\sigma(f_l)$. In the 'global' scheme, the minimum, mean, and standard deviation are computed over all fluxes (resp. all errors). In the 'per-filter' scheme, they are computed for each filter.

We use the Adam optimizer (Kingma & Ba 2014) to train our RNNs. Two learning rate policies were implemented: a standard 'decay on plateau' policy that decays the learning rate by a constant factor when the validation loss stagnates and the one-cycle policy introduced in Smith (2018).

Dropout is applied to all our models for regularization (except in the 'BBB' case where regularization is enforced by a Kullback Leibler divergence term in the loss function). Owing to the large size of the data set and limited overfitting observed, we did not use data augmentation. However, to improve performance with incomplete light curves, we found it beneficial to train the network with randomly truncated light curves. To make better use of parallelism,

[3] https://github.com/supernnova/SuperNNova

[4] In the github paper branch these methods are named MC dropout and Bayesian, respectively.

we batch together multiple light curves at a time, using zero-padding and masking where appropriate.

2.5 Random forest (RF)

To compare our RNNs with other classification methods, we implement an additional classifier. We use the excellent scikit-learn (Pedregosa et al. 2011) library to implement an RF classifier, which has been previously shown to have high performance in supernova photometric classification tasks (Lochner et al. 2016; Möller et al. 2016; Dai et al. 2018). Features for this classifier are extracted from complete light curves with the SALT2 fitter (Guy et al. 2007). This method has been shown to have good performance in binary classification with complete light curves (Lochner et al. 2016). We emphasize that the SALT2 fitter requires a redshift as input. Thus, even if no explicit redshift information is given to the RF classifier, the information implicitly percolates through the SALT2 fit parameters. We retained the following features to train our RF classifier: stretch ($x1$, $x1_{\mathrm{ERR}}$), colour (c, c_{ERR}), magnitude in the B band (mB, mB_{ERR}), normalization (x_0, $x_{0_{\mathrm{ERR}}}$), goodness of fit (χ^2), and observed magnitudes in the *rigz* bands ($m_{0\mathrm{obs}_r}$, $m_{0\mathrm{obs}_i}$, $m_{0\mathrm{obs}_g}$, $m_{0\mathrm{obs}_z}$, and their errors).

2.6 Convolutional neural networks (CNNs)

CNNs are another type of deep learning networks. They were originally introduced for image processing (Lecun et al. 1998). CNNs are made up of a cascade of convolutional layers that perform sliding cross-correlation of learned weight kernels with the input data. In astronomy, they have been successfully applied to various problems including the classification and discovery of astrophysical objects (Gieseke et al. 2017; Kimura et al. 2017; Carrasco-Davis et al. 2018; Lanusse et al. 2018). For a comparison with our RNN architecture introduced in Section 2.4, we implemented a simple CNN architecture in PyTorch (Paszke et al. 2017). Our network is made up of four 1D convolutional layers. Following Section 2.4, we collapse the sequence to a fixed-length representation and obtain the final prediction by applying a linear projection layer.

3 BASELINE RNN

We introduce a 'baseline RNN' that is trained in the standard, non-Bayesian way. In this section, we demonstrate the capabilities of this baseline model. We show that it is able to accurately classify light curves at each time-step as shown in Fig. 2. We study the impact on the performance when additional information such as host-galaxy redshift is added, when different training sets are used and when varying the number of classification targets (binomial, ternary, and seven-way classification).

3.1 Hyper-parameters

As shown in Section 2.4, different network architectures can be easily implemented in SuperNNova. We assess the impact of these variations using two metrics: the complete light-curve classification accuracy and the AUC for the binary classification task for 0.2 of the *SALT2-fitted* data set. We probed batch size (64, 128, 512, 1024), optimizer (Adam, RMSProp), dropout on recurrent layers (0.05, 0.1, 0.2), layer type (GRU, LSTM), bidirectional (True, False), the number of layers (1,2), rnn output (standard, mean pooling), random length (True, False), and the two learning policies. We also compared two ways to compress the hidden state sequence

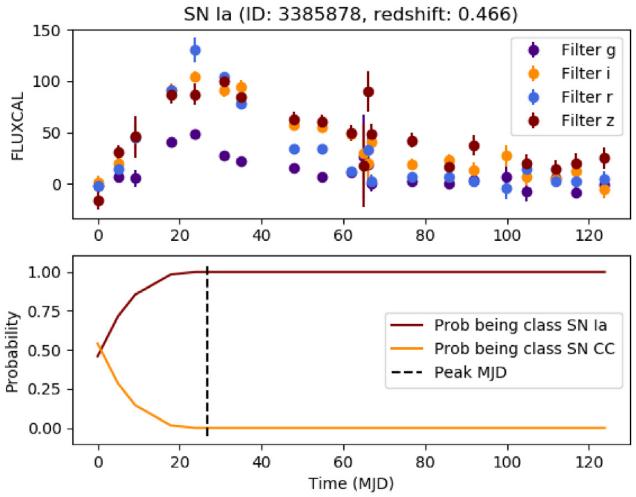

Figure 2. Simulated type Ia supernova light curve and classification. Top: Calibrated flux evolution in different DES band-passes as a function of normalized time (the first photometric measurement is set to time equal to zero). Bottom: Baseline RNN classification probability evolution with respect of time, no host-galaxy redshift information was provided. At each photometric measurement, classification probability is obtained. Maximum light of the simulated supernova is shown in a grey dashed line and simulated redshift of the supernovae is shown on the top, $z = 0.466$. We highlight that redshift is not used for this classification but can improve results. Our baseline RNN classifies this light curve as type Ia SN with great accuracy before maximum light; it only requires a handful of photometric epochs.

before the final prediction: last hidden state concatenation and mean pooling. These hyper parameters do not have a strong impact on the performance of the baseline RNN (classification accuracy for complete light-curve classification has a mean of 94.48 and a standard deviation of 1 per cent). We obtain two sets of hyper-parameters with the same accuracy of 96.01 without redshift information. We choose as our best hyper-parameters: batch size 128, Adam optimizer, 0.05 dropout on recurrent layers, bidirectional LSTM layers, 2 recurrent layers, cyclic learning rate, and standard rnn output. The latter however, is not preferred for partial light-curve classification, which will be explored in Section 3.5. To improve early light-curve classification, we choose the option of mean pooling before final prediction and training with randomly truncated light curves. These settings will be used throughout the following unless specified.

3.2 Normalization

Input feature normalization is a common practice to facilitate the training of neural networks. With half of the *SALT2-fitted* data set, we evaluate the impact of different normalization types for fluxes and errors: no normalization, a global normalization over all filters, and a per filter normalization. We find that the accuracy of our classifier is slightly improved when using global normalization (96.12 ± 0.07) when compared to no normalization (94.2 ± 0.2). We hypothesize that our use of optimizers with adaptive learning rate greatly mitigates feature scale issues. A per filter normalization is found to have equivalent performance (96.0 ± 0.2) to global normalization. Such a normalization scheme blurs the colour ratio between observed light curves in different band-passes. Interestingly, this indicates that colour ratios are not deemed an important feature for our algorithm's accurate classification of this supernova sample. An in-depth study of normalization impact is out of the

scope of this paper. Unless specified otherwise, we choose to apply global normalization throughout the rest of this work.

3.3 Comparison with other methods

The primary driver of supernova photometric classification has been to disentangle type Ia from other types of supernovae. We evaluate the performance of our SuperNNova baseline RNN on this task and compare it with different classification algorithms: the RF introduced in Section 2.5, the CNN structure in Section 2.6, and results obtained with another RNN architecture (Charnock & Moss 2017). We use the same training and validation samples from the *SALT2-fitted* data set to evaluate all algorithms. We assess uncertainties in the accuracy by performing five randomized runs that select different training, validation, and testing splits. We obtain for this data set an accuracy for the baseline RNN of 96.3 ± 0.4 and an AUC of 0.995 ± 0.001.

3.3.1 Baseline RNN versus CNN

As we did for our baseline RNN, we first choose the best hyper-parameters for the CNN. We probed batch size (64, 128, 512), random length (True, False), size of hidden dimension (16, 32, 64), and different learning rates (0.01, 0.001, 0.0001). Our best performance is found with: a batch size of 128, hidden dimension 32, learning rate 0.01, bidirectional layers, and training with random length sequences. For this configuration, we find that our CNN architecture obtains an accuracy of 94.53 ± 0.07 and an AUC of 0.9887 ± 0.0002 without any redshift information for the whole *SALT2-fitted data set*. Our baseline RNN obtains higher accuracies for this classification task. This behaviour is also seen with a smaller data fraction, 0.2 where classification accuracies are found to be 93.1 ± 0.2 (97.1 ± 0.1, 98.98 ± 0.07) without (with photometric, spectroscopic) host-redshift information. We note that other CNN configurations may yield better accuracies. Our CNN model is available in the SuperNNova repository[5] in order to foster further experiments on this classification task.

3.3.2 Baseline RNN versus RF

The RF classifier obtains accuracies of 95.15 ± 0.07 and an AUC of 0.9929 ± 0.0003 without redshift information. Our baseline RNN has a higher accuracy and AUC than many RF methods found in the literature (Lochner et al. 2016; Möller et al. 2016; Dai et al. 2018). However, classification accuracies depend on both the size and content of training and testing sets and therefore cannot be directly compared. We study the effect of the size of the training sample on the accuracy. As shown in Fig. 3, we find that our baseline RNN outperforms the RF for large data set sizes given the same redshift information. We also find that RNNs exhibit an increase in performance with larger data sets while the RF classifiers remain almost constant. Again, we stress that even when no explicit redshift information is provided to the RF classifier, the features obtained with the SALT2 fit implicitly contain this information. The high accuracy of our RNN shows that it has successfully learned meaningful representations on top of raw photometric time series, rather than using handcrafted features extracted from light-curve fitting. By using photometry directly, we avoid biasing our classified sample, a particularly relevant observation since the choice of feature extraction has a strong impact on performance (Lochner et al. 2016).

[5] https://github.com/supernnova/SuperNNova

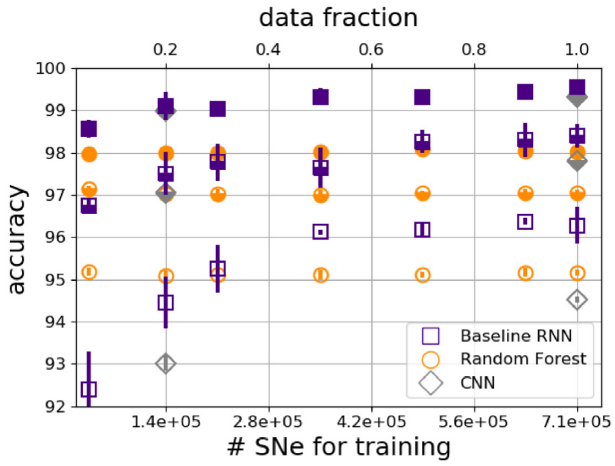

Figure 3. Ia versus non-Ia classification accuracy with respect to training set size using the *SALT2-fitted* data set. The top axis indicates the used data fraction and the bottom axis indicates the number of supernovae. The accuracy of baseline RNN is shown in indigo squares, CNN in grey diamonds, and RF in orange circles. Error bars are one standard deviation from the accuracy distribution for five runs with different random seeds to initialize the networks weights. Classification used: no host-galaxy redshifts are empty, photometric host-galaxy redshifts bottom filled, and spectroscopic host redshifts fully filled. Our baseline RNN achieves greater accuracies than the other classifiers for the same data set and explicit redshift information.

3.3.3 Baseline RNN versus another RNN architecture

For reference purposes, we compare a recently introduced RNN-based SN classifier with our baseline RNN trained with a similar size set. Charnock & Moss (2017) use half of the PCC data set 5 times data augmented, equivalent to ≈ 50 000 supernova light curves, and obtain an accuracy of 93.1 on binary classification without redshift information and 94.8 with redshift. By selecting a data fraction of 0.05 of our data set, we can train our baseline RNN with 32 222 light curves and obtain a classification accuracy of 92.4 ± 0.9 and an AUC of 0.980 ± 0.004 without redshift information; when using photometric redshifts, we obtain an accuracy of 96.8 ± 0.1. Although both methods were trained with different samples, our accuracies are comparable. Our algorithm seems to be more sensitive to redshift information, providing up to 4 per cent accuracy increase for this sample size.

3.4 Redshift, contamination, and efficiency

The addition of photometric redshifts, which are available for all our simulated supernovae, increases the accuracy of our baseline RNN by 2 per cent for photometric and 3 per cent for spectroscopic redshifts. Since spectroscopic redshifts are available for a subset of supernovae, we only evaluate performance on light curves for which this redshift is available. We will continue using this subsample performance throughout the rest of this manuscript.

A photometrically classified type Ia supernova sample is expected to have a small percentage of contamination from other supernova types. As mentioned in Section 2.1, our simulations are realistic and therefore include known SN rates and detection efficiencies for a survey such as DES. Therefore, our estimates are a good

indicator of the expected contamination by core-collapse SNe in a photometrically classified SN Ia sample.

We find that the contamination (or False Positive Rate) is dominated by types Ib and Ic supernovae when classifying without redshift information. This contamination peaks at simulated redshift between 0.2 for Ic and 0.4 for Ib and for our baseline RNN is of the order of a couple per cent for Ib and Ic SNe and under 1 per cent for other types. Contamination levels are found to be not correlated with the number of light curves for training. Within this 1 per cent contamination, there are type IIL and IIP SNe, which are more numerous than type Ic in our simulations but are mostly correctly classified by SuperNNova. Host-galaxy redshifts increase the performance of our classifier reducing contamination. In particular, simulated supernovae from Ib and Ic templates are better classified, reducing their contamination contribution in our baseline RNN from 2.5 ± 0.2 and 1.3 ± 0.1, respectively, to < 0.1 per cent with host redshifts. Interestingly, the classification of other core-collapse SNe such as IIP, IIn, and IIL1 is barely affected by this additional information.

Selection efficiency is an important metric when classifying type Ia supernovae. For spectroscopically selected samples, selection efficiencies drop quickly for faint events (D'Andrea et al. 2018; Kessler et al. 2018). Our baseline RNN performs superbly, with an almost constant efficiency as a function of simulated redshift with an average efficiency of 97.3 ± 0.4 without and up to 99.61 ± 0.09 with host-galaxy redshift information. Such high efficiencies enable probing new supernova populations at high redshifts and can have an important effect on selection biases found currently in cosmology as will be discussed in Section 5.

Furthermore, one of the biggest limitations of photometrically classified samples is the expected contamination level by other supernova types, which may affect statistical analyses (Hlozek et al. 2012; Jones et al. 2018). It has also been shown that for photometrically classified SNe Ia there is a compromise between sample purity and the efficiency of the classifier (Möller et al. 2016; Dai et al. 2018). We find that our highly efficient algorithm does not compromise the purity of the sample. On the contrary, the purity of an SN Ia sample classified without redshift is 95.4 ± 0.5, while the addition of redshifts increases the sample purity to 99.49 ± 0.05. This level of contamination, less than 1 per cent, is within the current range of contamination of spectroscopically classified samples currently used in cosmological and astrophysical analyses (Rubin et al. 2015).

3.5 Early light-curve classification

Having demonstrated the high performance of our baseline RNN in comparison to other classification methods, we now turn to explore other capabilities of SuperNNova. To do so, we use the *complete* data set that contains light curves that are not successfully fitted with SALT2. This increases the number and diversity of the training set for both core-collapse and type Ia supernovae. The effect of the SN diversity on the classifier performance is further studied in Section 5.2.

While there are currently a wealth of algorithms that can classify complete light curves, only a handful are able to classify partial light curves. Our RNN architecture allows us to accurately classify supernovae with a limited number of photometric epochs. At each light-curve time-step, as more information is added, the RNN hidden state is updated, allowing the network to adapt its predictions.

In Table 2, we present our baseline RNN accuracies for partial light-curve classification for the whole *SALT2-fitted* and *complete* data sets. The numbers of photometric epochs (nights) available at different stages of the light curves are shown in Fig. 4. In average, we require only ≈5 observing nights to obtain classification accuracies > 80 per cent. Since simulated light curves can contain photometric nights before the actual supernova explosion, we provide a 'realistic' estimate for unique nights useful in the classification. For this estimate, we select only photometric epochs within 14 d before peak maximum, which is a loose threshold for the rising time of type Ia supernovae. Using this more realistic metric, we require between 2.4 ± 1.2 and 3.3 ± 1.4 photometric epochs in average to start accurately classifying supernovae. We note that this low number of required epochs is linked to multicolour observations in each night, which provide valuable information for classification.

Table 2. Ia versus non-Ia classification accuracy using baseline RNN trained with whole *SALT2-fitted* data set, a fraction of 0.43, and the whole *complete* data set. Accuracy for partial light-curve classification with respect to days before or after simulated supernova peak, all indicates all data points available for each light curve. The addition of host redshift features is indicated as photometric (zpho) or spectroscopic (zspe). For the latter, since not all supernovae have a spectroscopic redshift, we show the accuracy of the subsample with spectroscopic host redshift.

Redshift	−2	0	+ 2	all
		SALT2-fitted data set		
None	83.6 ± 0.6	85.0 ± 0.7	86.4 ± 0.7	96.3 ± 0.4
zpho	93.2 ± 0.4	93.8 ± 0.5	94.5 ± 0.5	98.4 ± 0.3
zspe	97.0 ± 0.2	97.4 ± 0.2	97.9 ± 0.2	**99.55 ± 0.06**
		43 percent of complete data set		
None	86.1 ± 0.1	87.24 ± 0.08	88.31 ± 0.09	96.65 ± 0.05
zpho	93.0 ± 0.3	93.7 ± 0.3	94.3 ± 0.3	**98.6 ± 0.2**
zspe	92.7 ± 0.4	93.4 ± 0.4	94.0 ± 0.5	98.1 ± 0.2
		Complete data set		
None	86.4 ± 0.1	87.6 ± 0.1	88.6 ± 0.1	96.92 ± 0.09
zpho	93.5 ± 0.1	94.2 ± 0.1	94.8 ± 0.1	**98.85 ± 0.04**
zspe	93.3 ± 0.1	94.0 ± 0.1	94.6 ± 0.1	98.43 ± 0.08

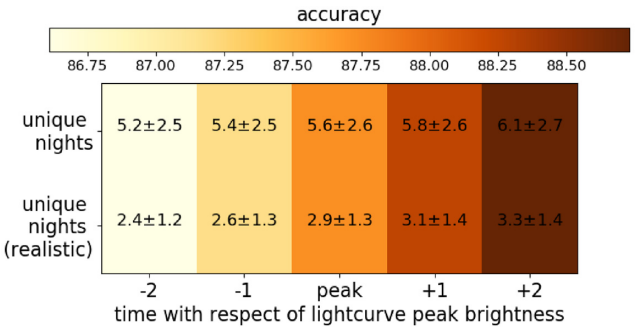

Figure 4. Average number of photometric epochs (nights) used for early light-curve classification. The rows indicate two ways of computing the average number of photometric nights available for classification while the columns indicate the days around simulated peak brightness of the supernova (−2, −1, at peak, +1, + 2). The raw average of photometric epochs is shown in the row 'unique nights'; since our simulated light curves can include observations before the supernova explosion, a more realistic estimation of the useful epochs include only observations two weeks before observed peak brightness, which is a loose estimate of the pre-maximum phase of a type Ia supernova light curve; this estimation is shown in the row 'unique nights (realistic)'. Colours indicate the accuracy obtained for these classifications with the baseline RNN as shown in Table 2 for the *complete* data set without any redshift information.

Additionally, since training set numbers affect performance metrics, we compare the performance with 43 per cent of the *complete* data set. This fraction of the complete data set has a similar size training set than the *SALT2-fitted* one.

For all data sets, our baseline RNN is capable of attaining a high-accuracy classification, $>$ 83 per cent right before the maximum of a supernova light curve. Further, we show that the addition of host-galaxy redshifts produces a rise in accuracy as high as 12 points.

We note that for spectroscopic redshifts, the *complete* sample has lower accuracy when compared with the use of photometric redshifts. This was not observed in the *SALT2-fitted* sample and it may be explained by a selection bias in the *complete* data set for supernovae with spectroscopic redshifts.

When classifying type Ia supernovae before or around maximum light, we find that contamination is still dominated by Ib supernovae with 9.4 ± 0.3 per cent contribution and Ic with 3.3 ± 0.2 per cent. Interestingly, type IIP and IIL2 supernovae can contribute around 2 per cent of the contamination each, while this is rarely the case for complete light-curve classification. This may be due to characteristic features of these light curves only present after maximum light, such as the plateau exhibited by type IIp SNe.

In summary, we have shown that SuperNNova is able to accurately classify light curves before and at maximum light. With accuracies ranging from 83.6 ± 0.6 up to 97.0 ± 0.2 for the *SALT-fitted* data set, without and with redshifts, respectively. SuperNNova opens a path towards an efficient use of photometric and spectroscopic resources for a follow-up. Candidates can then be prioritized for diverse science goals including targeted samples (e.g. SNe Ia for cosmology) and improving the SN sample for photometric classification as recently proposed by Ishida et al. (2018). Such a functionality will be crucial in the upcoming surveys where each night thousands of transients may be discovered.

3.6 Classifying many supernova types

There is more to supernova classification than binary classification. Time-domain surveys are increasingly exploring the diversity of supernovae and would benefit from the classification of multiple supernova classes. We explore ternary (Ia, Ibc, and IIs) and seven-way (Ia, IIP, IIn, IIL1, IIL2, Ib, and Ic) classification tasks. We train with the *complete* data set to obtain a large number of light curves per target.

For the ternary classification, we train with 318 820 light curves per type and for the seven-way classification with 104 158 per type. Accuracies for these classifications with and without redshifts are shown in Table 3. For complete light curves, our method yields an unprecedented classification accuracy, providing a useful tool for obtaining photometric samples of a diversity of supernovae. Early classification becomes a much more challenging tasks and we consequently observe a notable performance degradation. None the less, our algorithm provides a reasonable indication of the possible supernova type, and performance is enhanced with the incorporation of redshift information.

For an equivalent training sample per type, the ternary or seven-way classification accuracy with or without redshift and whole light curves is much lower than for binary classification (Ia versus non-Ia) as shown in Fig. 3. Splitting the core-collapse supernovae in subclasses adds a new level of complexity, which accounts for the performance drop. Interestingly, for the seven-way classification, the contamination of the SN Ia sample is dominated by type IIP SN as seen in Fig. 5. This was seen for early classification in the previous Section 3.5 but not for the complete light-curve classification.

Table 3. Ternary (Ia, Ibc, and IIs) and seven-way (Ia, IIP, IIn, IIL1, IIL2, Ib, and Ic) classifications using baseline RNN trained with complete data set. Accuracy for partial light-curve classification with respect to days before or after simulated supernova peak. The addition of host redshift features is indicated as photometric (zpho) or spectroscopic (zspe). We evaluate the accuracy for the complete validation sample and for the subset of light curves that possess a spectroscopic host-galaxy redshift.

Redshift	−2	0	+ 2	all
		Ternary classification		
None	69.2 ± 0.2	71.6 ± 0.3	74.1 ± 0.3	92.4 ± 0.3
zpho	79.7 ± 0.1	81.5 ± 0.2	83.4 ± 0.2	**95.4 ± 0.1**
zspe	77.8 ± 1.1	79.6 ± 1.2	81.5 ± 1.3	94.1 ± 1.0
		Seven-way classification		
None	57.2 ± 0.3	60.0 ± 0.4	62.9 ± 0.3	86.8 ± 0.3
zpho	64.5 ± 0.2	67.1 ± 0.2	69.8 ± 0.2	90.0 ± 0.1
zspe	64.2 ± 0.3	67.0 ± 0.4	69.8 ± 0.4	**90.4 ± 0.4**

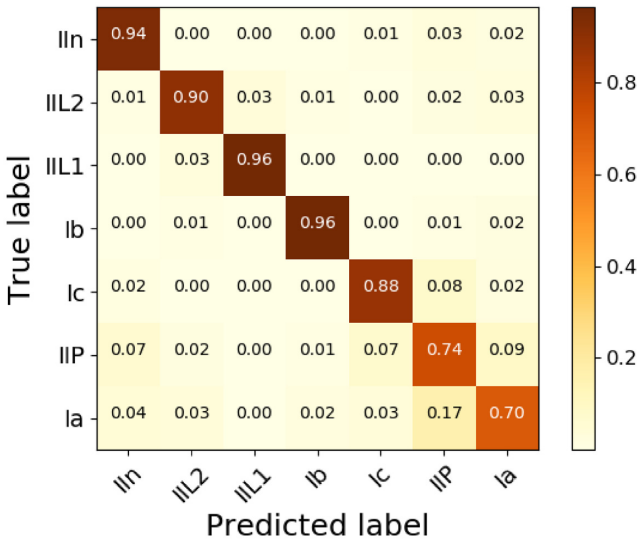

Figure 5. Confusion matrix for seven-way classification with best-performing baseline classifier with no redshift information. The matrix shows the fraction of light curves for a given supernova type (True label) classified as the supernova type in the horizontal axis (Predicted label). The diagonal elements are those light curves that are correctly classified, while off-diagonal elements are those that are mislabelled by the classifier. The colour bar indicates the normalized percentage of a certain type of SN light curves in the predicted label. For the seven-way classification, our algorithm performs superbly for types Ib, IIL1, and IIn SNe while SNe Ia are not as well characterized.

As seen in Table 3, early classification is severely impacted by adding more classification targets. A thorough study on mechanisms to improve multiclass predictions is out of the scope of this paper, but it is an interesting avenue for further studies. It is possible that a two-step procedure is useful, where: first, a multitarget prediction would identify the most probable targets and then, a second prediction with an algorithm specifically trained on the top candidates would refine the classification. The extensive results presented here show that SuperNNova can be a valuable tool for present and future surveys that wish to prioritize spectroscopic and photometric follow-up targets.

4 BAYESIAN RNNS (BRNNS)

In this section, we introduce Bayesian recurrent neural networks. These RNNs fit a posterior distribution on the weights of the neural network, thus allowing us to sample different predictions for a given input. Both of our BRNNs are derived from a technique called *variational inference*, which we will now quickly review.

4.1 Variational inference

Following Blundell et al. (2015), we can view neural networks as a model aiming to correctly estimate $\mathcal{P}(\mathbf{y}|\mathbf{x}, \mathbf{w})$, where, in our case, $\mathcal{P}$ is a categorical distribution, $\mathbf{y}$ is the classification target, $\mathbf{x}$ is the photometric light curve, and $\mathbf{w}$ are the network's weights. Neural networks are traditionally trained using a maximum likelihood criterion: Given a set of N labelled training observations, the data set $\mathcal{D}$ is defined by $\mathcal{D} = (\mathbf{x}_i, \mathbf{y}_i)_{i=1...N}$; we minimize the negative log likelihood $\mathrm{NLL} = \min_{\mathbf{w}} \sum_{i=1}^{N} -\log \mathcal{P}(\mathbf{y}_i|\mathbf{x}_i, \mathbf{w})$ with gradient descent.

Moving on to the Bayesian picture, instead of having a fixed value for each weight in the neural network, we now assign a *distribution* to each weight. We seek to find the *posterior* distribution of the weights: $\mathcal{P}(\mathbf{w}|\mathcal{D})$, which will then allow us to make predictions for new observations x: $\mathcal{P}(\hat{\mathbf{y}}|\mathbf{x}) = \int \mathcal{P}(\hat{\mathbf{y}}|\mathbf{x}, \mathbf{w})\mathcal{P}(\mathbf{w}|\mathcal{D})\,\mathrm{d}\mathbf{w}$. Typically, the *posterior* distribution is intractable for deep neural networks. To sidestep this difficulty, we can approximate the posterior with a simple parametric distribution $q(\mathbf{w}|\theta)$ called the *variational distribution*. In the Gaussian case, $\theta = (\mu, \sigma)$, respectively, the mean and the standard deviation of this variational distribution. Variational inference introduces a new learning criterion, where one seeks to minimize a divergence (typically, the Kullback–Leibler divergence or KL) between the *posterior distribution* and its *variational approximation*:

$$\hat{\theta} = \min_{\theta} \mathbf{KL}\left(q(\mathbf{w}|\theta)||\mathcal{P}(\mathbf{w}|\mathcal{D})\right) \tag{4}$$

$$= \min_{\theta} \left[\mathbf{KL}\left(q(\mathbf{w}|\theta)||\mathcal{P}(\mathbf{w})\right) - \int q(\mathbf{w}|\theta) \log \mathcal{P}(\mathcal{D}|\mathbf{w})\mathrm{d}\mathbf{w}\right] \tag{5}$$

$$= \min_{\theta} \left[\mathbf{KL}\left(q(\mathbf{w}|\theta)||\mathcal{P}(\mathbf{w})\right) - \mathbb{E}_{q(\mathbf{w}|\theta)}(\log \mathcal{P}(\mathcal{D}|\mathbf{w}))\right], \tag{6}$$

where $\mathcal{P}(\mathbf{w})$ is a user-specified *prior distribution* over the neural network's weights. This new cost function is made up of two terms. The first one, the KL term, is a regularization term: it penalizes variational distributions that differ too much from the prior. The second one is a likelihood term; our model must be flexible enough to handle the complexity of the data distribution. Bayesian optimization of neural networks is a trade-off between those two terms.

We will now review two ways in which the variational distribution can be specified, and investigate their applications to supernova cosmology. Results in the following sections use the *complete data set* for training, validation, and testing.

4.2 MC dropout

Following Gal & Ghahramani (2015a, b), we define our variational distribution to factorize over each row ∇ of the network's weight matrices: $q(\mathbf{w}_\mathrm{r}) = p\mathcal{N}(0, \sigma^2 I) + (1-d)\mathcal{N}(\mu_\mathrm{r}, \sigma^2 I)$, where $\mathbf{w}_\mathrm{r}$ is the weight matric rows, d is the dropout probability, $\mathcal{N}$ is the normal distribution, μ_r is the variational parameter (row vector) optimized with our gradient descent algorithm, σ^2 is a fixed, small constant, and I is the identity. Using a normal prior on the weights, the KL term can be approximated as L_2 regularization on the network's weights. Evaluating this network then becomes equivalent to performing dropout (i.e. masking with zeros) on the rows of the weight matrices (or equivalently, on each layer's input). The network can then be trained as usual, as long as we use the same dropout mask at every time-step in the sequence (Gal & Ghahramani 2015a, b).

We can now obtain a distribution of predictions, simply by sampling a different dropout mask for each prediction. In this work, we sample predictions 50 times for each light curve. The median of the prediction array is used to report the accuracy score. These uncertainties can provide valuable information in the classifier's confidence on the prediction.

We use the same hyper-parameters as those of the baseline RNN presented in Section 3. We probe different dropout values, $p \in [0.01, 0.05, 0.1, 0.2]$, and weight decay of the gradient descent optimizer $\in [1e^{-5}, 1e^{-7}, 1e^{-9}]$ to evaluate the performance of the network with a data fraction of 0.2. We find that the two most accurate architectures have a dropout of 0.01 and a weight decay of $1e^{-5}$ and $1e^{-7}$. Since accuracies for both models differ by less than 0.1 per cent, we choose to use weight decay $1e^{-7}$ in the following. For the *complete* data set, this configuration has an accuracy of 96.87 ± 0.09 without redshift and 98.8 ± 0.1 with host photometric redshift for complete light-curve classification, which is equivalent to the baseline RNN accuracy with a slight reduction. We find that the efficiency (98.91 ± 0.05), purity (98.6 ± 0.2), and contamination values are similar but slightly less than to those of the Baseline implementation. In the next Section 5, we will discuss other criteria for selecting the best-performing method, through calibration and entropy of OOD events.

4.3 Bayes by Backprop (BBB)

Following Fortunato et al. (2017), to model the variational distribution $q(\mathbf{w}|\theta)$, we use a simple Gaussian distribution with mean μ and standard deviation σ.

During training, we sample each weight $\mathbf{w}$ as follows: sample $\boldsymbol{\epsilon} \sim \mathcal{N}(0, 1)$ then obtain $\mathbf{w} = \boldsymbol{\mu} + \boldsymbol{\sigma} \times \boldsymbol{\epsilon}$, where $\boldsymbol{\sigma}$ and $\boldsymbol{\mu}$ are optimized by gradient descent.

Following Blundell et al. (2015), we partition the training data in n_B minibatches $\mathcal{D} = (\mathcal{D}_j)_{j=1...n_\mathrm{B}}$. For each batch, we optimize

$$\mathcal{L}_j = \frac{1}{n_\mathrm{B}} \mathbf{KL}\left(q(\mathbf{w}|\theta)||\mathcal{P}(\mathbf{w})\right) - \mathbb{E}_{q(\mathbf{w}|\theta)}(\log \mathcal{P}(\mathcal{D}_j|\mathbf{w})) \tag{7}$$

where the KL divergence is estimated with Monte Carlo sampling and the prior is given by a mixture of Gaussians with standard deviation σ_1, σ_2, respectively, as: $\mathcal{P}(\mathbf{w}) = \pi\mathcal{N}(0, \sigma_1) + (1-\pi)\mathcal{N}(0, \sigma_2)$. $\sum_{j=1}^{n_\mathrm{B}} \mathcal{L}_j$ is equivalent to the loss in equation (6).

Using the same hyper-parameters as those of the baseline RNN presented in Section 3, we study the impact of modifying the prior with 0.2 of the *SALT2-fitted* data set. We set $\pi = 0.75$. We searched $\log\sigma_1 \in \{-2, -1\}$, $\log\sigma_2 \in \{-4, -7\}$ for the recurrent layer and $\log\sigma_1 \in \{-2, -1, -0.5\}$, $\log\sigma_2 \in \{-1, -0.5, -0.1\}$ for the output layer. We initialized the standard deviation of the weights of the recurrent layer by sampling uniformly between $\log\left(\mathrm{e}^{\frac{\sigma_\mathrm{mix}}{\rho_\mathrm{lower}}} - 1\right)$ and $\log\left(\mathrm{e}^{\frac{\sigma_\mathrm{mix}}{\rho_\mathrm{upper}}} - 1\right)$, where $\sigma_\mathrm{mix} = \sqrt{\pi\sigma_1^2 + (1-\pi)\sigma_2^2}$, $\rho_\mathrm{lower} \in \{4, 3\}$ and $\rho_\mathrm{lower} \in \{2, 1\}$ for the recurrent layer, and $\rho_\mathrm{lower} \in \{3, 2, 1\}$ and $\rho_\mathrm{lower} \in \{2, 1\}$ for the output layer.

We find that the Bayesian network trains and attains better performance without the cyclic learning rate.

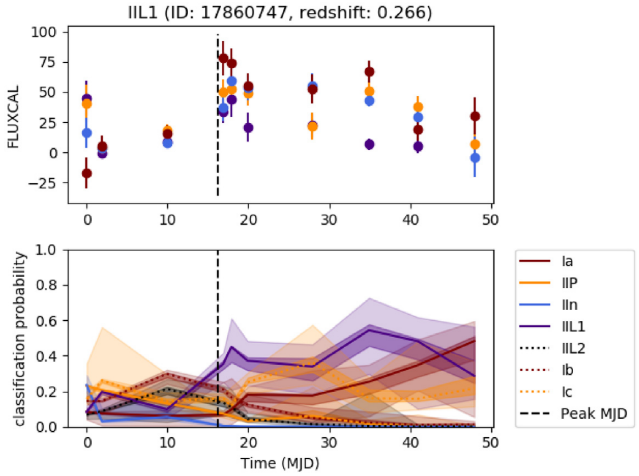

Figure 6. Classification of simulated type IIL1 SN as type Ia SN with seven-way MC dropout RNN. The model was trained without redshift information. Predictions for this light curve have low confidence and have large uncertainties (the shadowed regions show 68 and 95 per cent contours). The large uncertainties on the classification probability are representing the lack of confidence in this classification and are correlated to the low S/N.

The most accurate model has a $\log\sigma_1 = -1$, $\log\sigma_2 = -7$, $\rho_{\rm lower} = 4$, and $\rho_{\rm upper} = 3$ for the recurrent layer and $\log\sigma_1 = -0.5$, $\log\sigma_2 = -0.1$, $\rho_{\rm lower} = 3.0$, and $\rho_{\rm upper} = 2.0$ for the output layer. This model reaches an accuracy of 96.85 ± 0.05 without redshift and 98.72 ± 0.04 with host photometric redshift information for whole light-curve classification in the *complete* data set. The efficiency, purity, and contamination values are similar to those of the MC dropout implementation and slightly reduced with respect to the Baseline implementation ≤ 0.5 per cent. In Section 5, we will discuss other criteria for selecting the best-performing method, through calibration, entropy of OOD events, and model uncertainties.

4.4 BRNN uncertainties

In classification tasks, evaluating the reliability of a model's uncertainties is of paramount importance. Following Kendall & Gal (2017), we split uncertainties into two categories: *aleatoric* and *epistemic*. *Aleatoric* uncertainties capture the uncertainties in the input data, e.g. noise or other effects of data acquisition. *Epistemic* uncertainties express our ignorance about the model that generated the data. The latter depends on the network structure and the training set and therefore can be reduced with more flexible models, as well as larger and more diverse training sets.

In SuperNNova, we do not model *aleatoric* uncertainties since those are already included in the data set (through simulated flux measurement errors). By casting learning in a Bayesian framework, we expect to model *epistemic* uncertainties. An example of predictions for a light-curve with a Bayesian fremwork can be seen in Fig. 6.

For BRNNs, given a light curve $\mathbf{x}$, we can obtain a set of predictions $p_j(\mathbf{x})_{j=1...n_s}$ where the weights are sampled n_s times from the prediction's posterior. We compute the model uncertainty for a given light curve $\mathbf{x}_i$ as

$$\widehat{\sigma} = \text{std}\left(\sum_{j=1}^{n_s} p_j(\mathbf{x}_i)\right), \tag{8}$$

where $j \in [1, n_s]$ is the index of inference samples, $p_j(\mathbf{x_i})$ is the classification probability for the given light curve $\mathbf{x}_i$ for each inference sample j, and std is the standard deviation.

4.4.1 *Uncertainty evaluation metrics*

To evaluate the uncertainty estimates of the BRNNs in SuperNNova, we make use of two metrics:

(i) *Mean model uncertainty*: For a given test set $\mathcal{D}_k$, we average the model uncertainties ($\widehat{\sigma}$ from equation 8) over all light curves:

$$\langle\widehat{\sigma}_k\rangle = \frac{1}{N}\sum_{i=1}^{N}\widehat{\sigma}_i, \tag{9}$$

where $i \in [1, N]$ is the index of each light curve in test set $\mathcal{D}_k$.
For two given sets of predictions, we can compute the difference between their uncertainties as

$$\Delta\langle\widehat{\sigma}_{1,2}\rangle = \langle\widehat{\sigma}_1\rangle - \langle\widehat{\sigma}_2\rangle, \tag{10}$$

where $\widehat{\sigma}_i$ is defined in equation (9) for a set of predictions i.

(ii) *Entropy*: We follow Fortunato et al. (2017) and define, for a test data set $\mathcal{D}_t : [\mathbf{x}_1, ...\mathbf{x}_N]$ with N light curves and a classification model m, the entropy of $\mathcal{D}_t$ under m as

$$H_m[\mathcal{D}_t] = \sum_{i=1}^{N} p_m(\mathbf{y}_i|\mathbf{x}_i)\log\left(\frac{1}{p_m(\mathbf{y}_i|\mathbf{x}_i)}\right), \tag{11}$$

where $p_m(\mathbf{y}_i|\mathbf{x}_i)$ is the classification probability given the light curve i using model m. Entropy is a proxy for the model's confidence on predictions. Thus, confident predictions will yield low entropy. An equivalent per light-curve entropy can be computed as $\overline{H_m}[\mathcal{D}_t] = \frac{1}{N}H_m[\mathcal{D}_t]$, where N is the number of light curves in a given test set. For two given sets of predictions, we can define the *entropy gap* ΔH by

$$\Delta H = \quad \overline{H_{m_1}}[\mathcal{D}_t] - \overline{H_{m_2}}[\mathcal{D}_t] \quad or = \overline{H_m}[\mathcal{D}_{t_1}] - \overline{H_m}[\mathcal{D}_{t_2}], \tag{12}$$

where the first option evaluates the entropy gap over the same data set for two given models (m_1, m_2), and the second uses the same model to make predictions for two different data sets $(\mathcal{D}_{t_1}, \mathcal{D}_{t_2})$.

4.4.2 *Uncertainty evaluation*

We now evaluate the uncertainty estimates of the BRNNs in SuperNNova. Bayesian neural networks aim to capture *epistemic uncertainties* by putting a prior distribution over the NN's weights. Bayesian inference leads to computing a posterior that represents the set of plausible models, given the data. As more data are available, we expect these uncertainties to be explained away. To verify this, we make predictions with two models on the same test set. The models differ only by the number of light curves used for training.

First, we compute $\Delta\langle\widehat{\sigma}\rangle$ (equation 10) between the predictions obtained with a model trained with a small number of light curves and one trained with a larger number. We expect this metric to be positive. For the *complete* data set, we compute the $\Delta\langle\widehat{\sigma}\rangle$ of models trained with 43 per cent and the whole training set, $\Delta\langle\widehat{\sigma}\rangle = \langle\widehat{\sigma}_{0.43}\rangle - \langle\widehat{\sigma}_{\rm whole}\rangle$. We find $\Delta\langle\widehat{\sigma}\rangle = 0.005$ and 0.007, respectively. We find similar values for the $\Delta\langle\widehat{\sigma}\rangle$ of models trained with the half and *SALT-fitted* training sets.

Secondly, we compute the entropy gap defined in equation (12) between the same predictions. Since we expect the predictions from the model trained with less light curves to be more uncertain than the one from a model trained with a larger data set, we should obtain a

positive ΔH. We find for the MC dropout and BBB implementations a positive $\Delta H > 0.01$.

Both BRNN implementations have uncertainties that are consistent with the behaviour expected of *epistemic* uncertainties as shown by the metrics computed above. However, we find that the size of uncertainties differs in both methods. If we compute the mean of the classification uncertainties when classifying the *complete* data set, we find that the MC dropout implementation has uncertainties twice larger than the BBB implementation. This may be due to initialization effects or more fundamental effects due to the way each method specifies variational distributions. Future research should strive to improve the comprehension of these uncertainty estimates.

In the following section, we will further study the behaviour of our BRNNs uncertainties. In particular, we will explore the effect of non-representative training sets and the classification of OOD light curves.

5 TOWARDS COSMOLOGY AND OTHER STATISTICAL ANALYSES

To perform statistical analyses using photometrically classified supernovae, a high-accuracy algorithm is not enough. It is equally important to show that it is statistically sound. By that, we mean that it should provide well-calibrated probabilities and capture the *epistemic* uncertainties related to the classification model. In the following, we will quantify the performance of our algorithms with respect to these requirements focusing on the Ia versus non-Ia classification tasks.

5.1 Calibration

Classification probabilities should reflect the real likelihood of events being correctly assigned to a target. Classification algorithms where this is true are said to be calibrated. Niculescu-Mizil & Caruana (2005) show that common machine learning algorithms such as SVMs and boosted trees do not predict well-calibrated probabilities, pushing predicted probabilities away from 0 and 1. For other algorithms such as RF, the calibration is heavily data dependent. Recently, Guo et al. (2017) showed that modern, deep neural networks also suffer from poor calibration and that there is a trade-off between classification performance and calibration.

To analyse our algorithms' calibration, we use reliability diagrams (DeGroot & Fienberg 1983). These diagrams are constructed by discretizing the predicted probability into 10 evenly spaced bins. A predicted probability between 0.0 and 0.1 falls into the first bin, and so on. For each bin, we plot the fraction of true positive cases against the mean predicted probability in that bin. The fraction of true positive cases in the binary classification is defined by the number of type Ia supernovae in that probability bin with respect to all supernovae in that bin. If the model is well calibrated, the points will fall near the diagonal line. This is equivalent to saying that in a sample with a hundred events classified as type Ia with a probability 0.7, we expect 70 per cent of events to be true SNe Ia and 30 per cent to be misclassified SNe from other types. Furthermore, we construct a metric to study the calibration deviation: the difference between the two calibrations squared.

For the RF algorithm, we find a large calibration deviation when classifying the *SALT2-fitted data set*, as can be seen in Fig. 7. Over five randomized runs, the level of dispersion is found to be 0.025 ± 0.002. In this classification task, RNNs are found to have better calibration than the RF algorithm with a dispersion an order of magnitude lower. For BBB and MC dropout RNNs, we construct

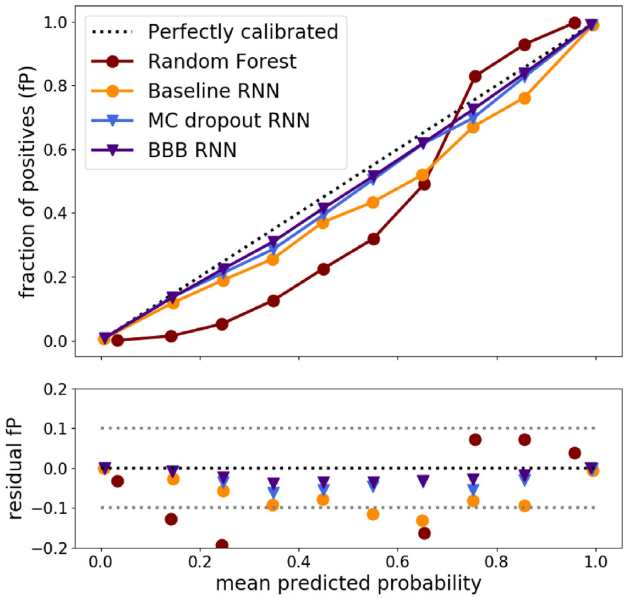

Figure 7. Calibration of classification algorithms. Top: reliability diagram showing the calibration for *SALT2-fitted* data set classification for a single seed. We use the most accurate configurations for the RF (red circles), baseline RNN (yellow circles), MC dropout RNN (blue triangles), and BBB RNNs (purple triangles). Bottom: dispersion from perfectly calibrated algorithms. Note that the RF algorithm has a large deviation from perfect calibration while the RNNs are better calibrated than this algorithm with the BBB implementation almost perfectly calibrated.

reliability diagrams using multiple predictions per sample, rather than the median prediction. We find that diagrams built this way exhibit better calibration than those built with a single prediction per sample. This is evidence that the model has learned meaningful predictive uncertainties since including the complete distribution of probabilities improves the calibration.

Bayesian RNNs are found to be better calibrated than the baseline RNN for both *SALT-fitted* and *complete* data sets. For the classification of the *complete* data set without any redshift information, we find a calibration dispersion from the baseline RNN of 0.006 ± 0.001, which is reduced to 0.004 ± 0.002 for the MC dropout and to 0.0005 ± 0.0004 for the BBB implementations.

Calibration depends on the nature and size of the training set. We verify this, by measuring the dispersion for the baseline RNN when classifying the *SALT2-fitted data set* without redshift information with data fractions between $\{0.2–1.0\}$. For the nature of the training set, we compare using the whole *SALT2-fitted* data set and 0.43 of the *complete* data set. We find that the data fraction or nature of the data set can change the dispersion up to 50 per cent.

Photometrically classified samples are usually selected from those events that have a probability larger than a given threshold. These thresholds are chosen as a compromise between purity and size of the selected sample. However, miscalibration affects the positive fraction of events in each bin, providing misleading probabilities. To account for large calibration deviation, two approaches may be taken: either to perform a post-processing recalibration (e.g. Niculescu-Mizil & Caruana 2005; Guo et al. 2017) or the difference between the obtained and true probability for each bin can be used to re-weight obtained probabilities. This will be of importance for classifier and data sets where large calibration dispersion is observed. We consider our BBB and MC dropout RNNs to be well calibrated due to deviations less than 1 per cent.

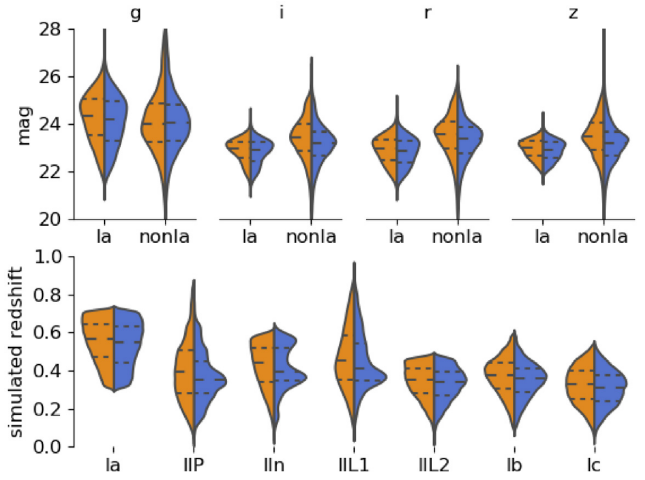

Figure 8. Distributions of maximum observed brightness (mag), in all DES filters (g, i, r, z), and of simulated redshift for *SALT2-fitted* (left yellow) and *complete* (right blue) data sets. Maximum observed brightness is shown for type Ia and non-Ia samples while simulated redshift is shown for each of the available templates. The dashed lines show the median and first quartile of the distribution. The *complete* and *SALT2-fitted* data sets probe similar parameter space but have different distributions. This is similar to what is expected of non-representative samples.

5.2 Representativeness

To improve generalization, classification algorithms typically require large and representative training sets. A training set is called representative if its properties, such as maximum observed brightness, have distributions that resemble the distributions of the data set to classify also known as a test set. Training sets are usually simulated using templates from spectroscopically followed up supernovae. Current and past surveys have spectroscopic follow-up strategies designed to target as many Ia-like objects as possible. These spectroscopically selected samples are therefore not probing the complete supernova diversity. In this section, we investigate the impact of training SuperNNova with non-representative and representative data sets.

Previous works have discussed the issue of representativeness by comparing the accuracy of an algorithm trained with a data set whose size was similar to the test set (Lochner et al. 2016; Charnock & Moss 2017; Pasquet et al. 2019). RNN-based methods, trained on small data sets, are prone to overfitting (Charnock & Moss 2017). Additionally, we have shown that the performance of our algorithm depends on the size of the training set (see Fig. 3). Thus, it would not be adequate to use this procedure to evaluate the impact of representativeness.

Instead, we use the following approach. We select the *SALT2-fitted* data set as an example of non-representative data. Recall that this data set contains light curves successfully fitted by SALT2 and is therefore biased towards light curves that look like SNe Ia. As representative data we select the *complete* data set that includes all simulated light curves 'discovered' by the DES-like survey and pass selection cuts. This sample has larger diversity of light curves and is thus a better probe of the full data distribution space. The distribution of magnitudes and redshifts for both data sets is shown in Fig. 8. To investigate the discrepancy between spectroscopic and photometric samples, we train with the non-representative data set and evaluate the classification performance for the test sample in the representative data set.

For classification with no redshifts using the Baseline and Bayesian RNNs, we find that the accuracy is reduced by 1 per cent when trained with a non-representative data set. Although small, this variation is not within the uncertainties of our model accuracies.

As discussed in Section 4.4, Bayesian RNNs can capture *epistemic uncertainty*, which includes the lack of diversity in the model's training set. Therefore, we expect a Bayesian RNN trained on a non-representative set (in our case, the full *SALT2-fitted* data set) to be more uncertain than one trained on a representative set (in our case, a subset of the *complete* data set) when evaluating on said representative set. To quantify this in a rigorous way, we use the two metrics introduced in Section 4.4.1. We find both the metrics to be positive for all BRNNs, $\Delta\langle\widehat{\sigma}\rangle > 0.004$ and $\Delta H > 0.01$.

The lack of representativity and the limitations of supernova templates are major issues in SN photometric classification. Recently, Ishida et al. (2018) introduced a framework for the optimization of spectroscopic follow-up resources to improve SN photometric classification data sets. Bayesian RNN uncertainties may be a promising indicator to select follow-up candidates for this purpose. This is an interesting possibility that we leave for future work.

5.3 OOD light curves

In astronomy, as in any other classification application, the generalization properties of a classifier and its behaviour on unseen, possibly OOD samples represent a challenge. In this section, we study the performance of SuperNNova when classifying OOD light curves. We test four different types of OODs: time-reversed light curves, randomly shuffled light curves, random fluxes, and a sinusoidal signal. The latter two were generated using the same cadence and flux range as normal supernovae. These light curves are only used for testing and were not used for training at any time.

When classifying OOD light curves, all SuperNNova algorithms rarely classify these light curves as SNe Ia. For binary classification, for the MC dropout and BBB implementations, the reverse and shuffled light curves obtain the largest number of classifications as SNe Ia, 4.8 per cent and 4.1 per cent, respectively, with the MC dropout implementation and less than 3 per cent for the BBB implementation. Many of these light curves resemble supernovae, specially with light curves with a low signal-to-noise ratio. For the sinusoidal light curves, only the MC dropout classifier classifies a significant percentage as SNe Ia 5.9 per cent, while for the BBB method this is not the case. The network seems to characterize type Ia supernovae well and therefore classifies most OOD events as core-collapse supernovae as can be seen in Fig. 9 with the BBB RNN. In ternary and seven-way classifications, the most common predictions for OOD events are types II: IIn, IIp, IIL1.

To assess how our prediction uncertainties behave with respect to OOD events, we compute ΔH for our best-performing models comparing OOD and our *complete* data test set predictions. Although ΔH does not measure uncertainty in an absolute way, it can be used as a qualitative test where OOD events should show high entropy as seen in Fortunato et al. (2017). For the binary classification problem, we find positive ΔH for random and reverse light curves for both the MC dropout and BBB implementations, with the BBB having the largest entropy gap. Interestingly, for sinusoidal and shuffled light curves, this metric is negative for the MC dropout implementation. For the ternary and seven-way problems, other OOD predictions are negative as well. While Fortunato et al. (2017) observe large positive entropy for OOD events, our experiments show surprisingly a mixed behaviour. We explore this question in Appendix B where we conclude that RNNs are at risk of collapse

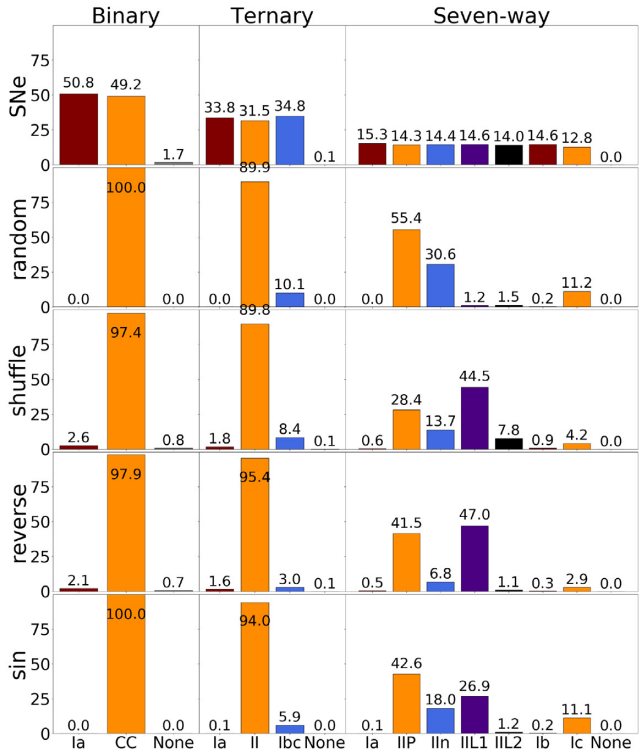

Figure 9. Predicted classification percentages for best-configuration BBB RNN with single seed. The columns indicate the number of targets for the classification (with probabilities summing one) and an additional column indicating a low probability prediction, <0.6, 0.4, and 0.2 for 2, 3, and 7 classes, respectively. The rows represent the different light-curve types. All the rows are OOD-generated light curves except the row 'SNe' representing our testing data set. Since the testing set has balanced classes, we expect the prediction percentage in the row 'SNe' to be balanced as well. Note that the OOD events are rarely classified as type Ia SNe (< 2.6 per cent binary, < 1.8 per cent ternary, and < 0.6 per cent in seven-way classification); the highest percentages in binary classification are for OOD, which can resemble SN light curves, e.g. reverse and shuffled. In ternary and seven-way classifications, OOD events are preferably classified as type II, IIn or IIL1 SNe. Note that the number of low probability detections is much higher for OOD when classifying in three or seven classes.

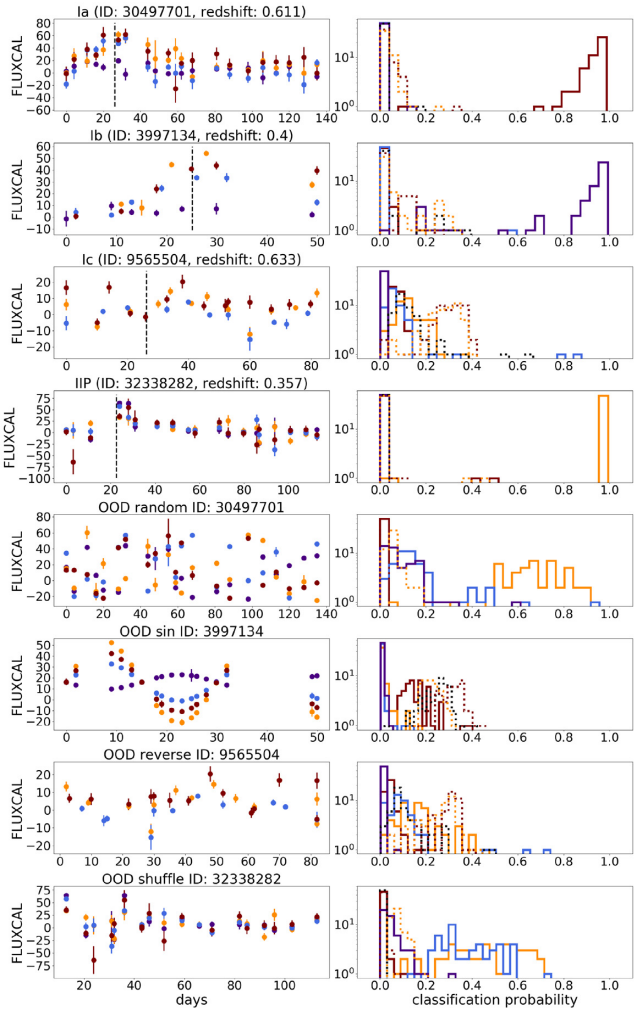

Figure 10. OOD and SN light-curve prediction distribution for best MC dropout configuration without redshift information for seven-way classification. The first column depicts light curves, and the second column their classification probabilities for different supernova classes depicted with various line styles and colours. The first four rows show SN light curves while the last four rows are OOD events. For the OOD events, we see different behaviours: high classification probabilities for a certain supernova class with large uncertainties, clustered classification probabilities around a small value, and a mixture of both behaviours.

on predicting a single class with high probability. Additionally, we show the different behaviours for the classification of OOD events in a seven-way classification in Fig. 10.

5.4 Type Ia supernova cosmology

Current and future surveys such as DES, PanSTARRS, and LSST will use photometrically classified type Ia supernovae for their cosmological analyses. In this section, we examine the properties of type Ia supernova samples classified by SuperNNova.

To study of the expansion of the Universe, type Ia supernovae are used to measure distance modulus as a function of redshift. We can compute the distance modulus, μ, of a given type Ia supernova:

$$\mu = m_B + \alpha x_1 - \beta c + M, \tag{13}$$

where M is the absolute magnitude of the SN Ia, m_B is the rest-frame B magnitude (at peak luminosity), x_1 is the stretch parameter, and c is the colour. The last three parameters are derived from the SALT2 fit to the observed SN Ia light curve. M, α, and β and can be constrained during cosmological fits. In this work, we will use the simulated values: $M = 19.365$, $\alpha = 0.144$, and $\beta = 3.1$.

With SuperNNova, we obtain a photometrically classified type Ia supernova sample by selecting light curves that have a probability larger than 0.5 to be type Ia. Importantly, we have shown in Section 5.3 that OOD light curves are rarely classified as type Ia SNe for any of our algorithms. We do not correct calibration deviations found in Section 5.1 given the small miscalibration found, which will have a minor effect on the photometrically selected SN Ia sample population, which is assessed in this section.

For each supernova in the photometrically selected sample, we obtain its distance modulus using equation (13) and compare it with the ΛCDM distance modulus corresponding to that simulated redshift. This is called the Hubble Residual and ideally should be around zero. We show the Hubble Residual for a sample classified with the best-performing BBB RNN without redshift information in Fig. 11 for both correctly classified type Ia supernovae and

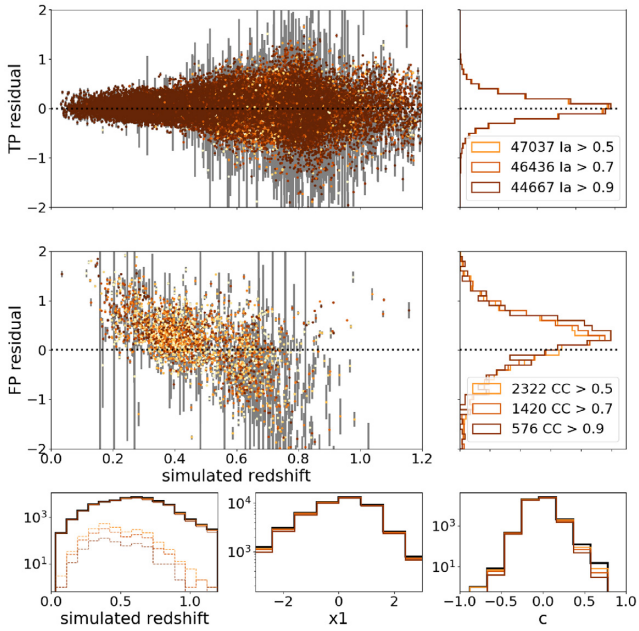

Figure 11. Hubble Residuals as a function of simulated redshift for the photometrically classified SN Ia sample using our best-performing BBB RNN trained with the *complete* data set with no redshift information. Top left: Hubble Residuals for correctly classified type Ia supernovae (TP) as a function of simulate redshift. Colours correspond to classification probabilities smoothly distributed between 0.5 (yellow) and 0.9 (dark orange). Top right: normalized histogram of Hubble Residual for three selection thresholds: 0.5 (yellow), 0.7 (orange), and 0.9 (dark orange). Centre left and right: same plots as above but for incorrectly classified core collapse, which represent the sample contamination. Bottom from left to right: redshift, colour (c), and stretch (*x1*) distributions of the complete simulated Ia sample (black), selected TP at different probability thresholds (coloured continuous lines) and selected FP (dashed lines). We plot only the supernovae classified as type Ia that have had a successful SALT2 fit, which provides parameters necessary to compute the distance modulus. Note the small difference between the simulated and selected redshift, colour, and stretch distributions and the contamination contribution < 5 per cent. Clearly, such a photometric sample significantly reduces selection biases usually seen in spectroscopically selected samples.

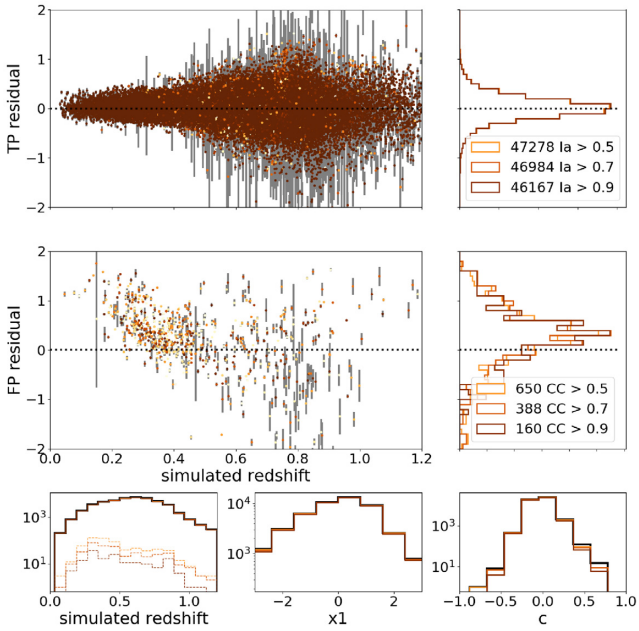

Figure 12. Hubble Residuals as a function of simulated redshift for the photometrically classified SN Ia sample using our best-performing BBB RNN with photometric host-galaxy redshift information. The figure is similar to Fig. 11. Note that the level of contamination obtained in this sample is less than 2 per cent, which is equivalent to that expected of spectroscopically classified samples. As in the previous figure, our selected sample probes almost completely the whole redshift, colour, and stretch parent population.

incorrectly classified core-collapse supernovae. The contribution by the latter represents less than 5 per cent of the sample and peaks at redshifts between 0.4 and 0.5.

Such contamination can be reduced by: using additional information such as host-galaxy redshifts, raising the probability threshold for sample selection as shown in Fig. 11, or eliminating classified events with large uncertainties. We have shown that adding redshifts allows us to obtain a sample contaminated by < 2 per cent core-collapse SNe, contamination that is within the range of current spectroscopically classified samples (Rubin et al. 2015). Raising the probability threshold is found to be a good alternative, e.g. for the MC dropout implementation, setting the threshold to >0.9 increases the sample purity by reducing the number of misclassified core-collapse SNe by a factor of 4 and only reducing the correctly classified SN Ia by < 5 per cent.

Uncertainties from Bayesian RNN's may provide useful information to discard supernovae in cosmology analyses. They may be incorporated to the analysis of photometric samples with Bayesian Hierarchical Cosmology models such as Steve (Hinton et al. 2018). As a first toy approach, we discard those SNe classified with uncertainties larger than 0.05 in the Hubble Diagram, finding a similar decrease in the number of core-collapse SNe as raising the threshold to 0.9. The numbers of TP SNe Ia are also equivalent. This shows that uncertainties are a good indicator for possible contamination. A more robust study on the impact of selection by classification probabilities and uncertainties is left for future work.

Finally, we examine the population properties of our SuperNNova-selected sample when compared to its parent population as can be seen in Figs 11 and 12. In spectroscopically selected samples, there is a large difference between its properties and the ones of the expected supernova population. This is mainly due to spectroscopic selection effects (D'Andrea et al. 2018; Kessler et al. 2018). In our photometrically classified sample using no redshift information, we see an almost constant selection efficiency throughout redshift. Importantly, colour and stretch distributions match very closely the parent population. Aside from this selection bias reduction, we are able to classify type Ia supernovae at redshifts >0.9, which is extremely difficult when using spectroscopy.

6 SUPERNOVA RESOURCES

SuperNNova has been optimized for a reasonable computing budget and special care has been dedicated to memory consumption. For this project, we used an eight-core Intel(R) Core(TM) i7-7700K CPU (4.20GHz) with 15 GB of RAM and a NVIDIA GeForce 1080Ti GPU to speed up training and inference.

Data base construction, which includes formatting and normalization, is carried out once for the whole data set and requires less than half an hour. For a given model, training can be carried out on the CPU or, if available, on the GPU. We quote speeds for a batch size of 128, larger batch sizes will result in a decrease of training time. Similarly, we quote times for training with cyclic learning

Table 4. Classification speeds for batch sizes of 128 light curves. For each RNN model, we measure how many supernovae per second can be classified using either CPU or GPU.

RNN model	The number of SNe per second	Device
Baseline	23 280	GPU
Baseline	2820	CPU
MC dropout	342	GPU
MC dropout	58	CPU
BBB	264	GPU
BBB	56	CPU

rate, without cyclic training times increase. For the baseline RNN, we find that training a single epoch with approximately 5000 light curves ranges from 1 min (GPU) to 10 min (CPU). Training the baseline RNN to convergence requires from one and a half hours (GPU) to 15 h (CPU). For the same light-curve numbers per epoch, MC dropout RNNs require slightly longer to train with 1.5 min (GPU), while BBB RNNs require 2 min per epoch (GPU).

Once a model is trained, it can be applied to classify independent light-curve samples. SuperNNova is able to classify tenths to hundreds of light curves per second with BRNNs as can be seen in Table 4. This includes sampling 50 times the probability distribution (50 being the number of inference samples). For the baseline RNN (using either GPU and CPU), we are able to classify thousands of light curves per second. Inference speed will be critical given the unprecedented amounts of data expected from upcoming transient surveys. Thus, SuperNNova paves the way towards a real-time light-curve classification.

7 DISCUSSION AND CONCLUSIONS

We have presented SuperNNova, a fast and accurate framework for photometric classification. SuperNNova only requires light curves as input. Additional information, such as host-galaxy redshifts, can be easily incorporated to improve its performance. We have released the source code on github alongside with extensive documentation, tests sets, and a docker environment to foster exploration and reproducibility.[6]

Current and future surveys will continue to discover a larger number of transients than they are able to spectroscopically characterize. Photometric classification will be key to harness the full power of these surveys, both for optimizing follow-up resources and for obtaining larger and more diverse samples than those that can be spectroscopically classified.

SuperNNova is able to achieve large and high-purity SN samples. We have shown that it has high performance in the classification of type Ia versus non-type Ia supernovae with accuracies up to 95 per cent for whole light curves. When host-galaxy information is provided, accuracies increase to 98 per cent. SuperNNova can be used in various classification tasks such as ternary and seven-way classifications. For these classification tasks, accuracies are found to be above 90 per cent when host-galaxy information is provided. By selecting photometric samples, SuperNNova allows the exploration and characterization of diverse astrophysical objects.

[6]https://github.com/supernnova/SuperNNova

With the advent of large surveys discovering thousands of transients every night, it will be imperative to prioritize a follow-up using partial light-curve classification. SuperNNova achieves an unprecedented accuracy on this particularly challenging task. Our results show that we are capable of discriminating Ia and non-Ia supernova light curves before their maximum peak with accuracies up to 86 per cent without redshift information and higher than 90 per cent when host-galaxy redshifts are provided. Classification becomes harder when classifying into many targets (3–7), with accuracies only reaching around 63 per cent near maximum light. For this classification task, the addition of host-galaxy redshifts can yield accuracy improvements as high as 10 points. Besides accuracy, inference speed is another key concern. With SuperNNova, we set a strong baseline: Our algorithm is capable of classifying thousands of light curves per second.

Neural networks and other machine learning algorithms have been successfully applied to astrophysical analyses but their statistical properties have rarely been analysed in detail. In this work, we devise tests to estimate the robustness of classification algorithms. We show that calibration depends on the choice of algorithm and data set and emphasize the need for calibration diagnosis. We find that our RNNs are better calibrated than an RF algorithm applied to the same data set.

In SuperNNova, we adopt a Bayesian view of neural networks, showing much improved uncertainty estimates, calibration, and out-of-sample behaviour over standard neural networks. We have shown that the uncertainties obtained with our BRNNs reflect the confidence of predictions and exhibit the behaviour expected for epistemic uncertainties.

One of the main goals of SuperNNova is to obtain photometrically classified type Ia supernovae. Current large surveys such as DES and PanSTARRS are already exploring methods to obtain cosmological constraints from such samples (Jones et al. 2018). Accounting for the effect of large contamination expected in photometric samples is one of the major limitations of these analyses. With SuperNNova, we can achieve a contamination of < 2 per cent, equivalent to that of current spectroscopically classified samples (Rubin et al. 2015).

Furthermore, using photometrically classified supernovae in statistical analyses requires a thorough understanding of classification reliability and the limitations of the available training samples. We have shown that our Bayesian RNNs are reliable and well calibrated, and provide meaningful uncertainty estimates. These uncertainties can be used in existing cosmology frameworks to obtain cosmological constraints (Hinton et al. 2018) or follow-up optimization frameworks to improve supernova templates (Ishida et al. 2018).

We have made initial assessments on the selection biases in a photometrically classified type Ia supernova sample with SuperNNova, finding a higher selection efficiency and a better population sampling when compared with spectroscopically classified supernovae in the recent DES analysis (Kessler et al. 2018). Importantly, we have shown that OOD, random, and sinusoidal-shaped light curves are mostly rejected from our photometrically classified SN Ia sample.

SuperNNova has been designed with supernova light-curve classification in mind; however, it can be easily adapted to other time-domain astronomy classification problems, e.g. the classification of other transients such as kilonovae, variable stars, and a wider variety of supernova classes. We expect to continue applying SuperNNova and improving it to a wide variety of classification problems.

ACKNOWLEDGEMENTS

We thank Richard Scalzo and Chris Lidman for their feedback on this manuscript, Samuel Hinton for testing the framework, Rick Kessler for help with simulations, and Alex Kim, Wade Blanchard, and Robert Clark for useful statistics discussions.

Parts of this research were conducted by the Australian Research Council Centre of Excellence for All-sky Astrophysics (CAASTRO), through project number CE110001020.

REFERENCES

Astier P. et al., 2006, A&A, 447, 31
Astropy Collaboration, 2013, A&A, 558, A33
Bahdanau D., Cho K., Bengio Y., 2014, preprint (arXiv:1409.0473)
Bazin G. et al., 2011, A&A, 534, A43
Bellm E. C., 2019, Life beyond PTF, Southern Horizons in Time-Domain Astronomy, p. 160.
Bernstein J. P. et al., 2012, ApJ, 753, 152
Betoule M. et al., 2014, A&A, 568, A22
Blundell C., Cornebise J., Kavukcuoglu K., Wierstra D., 2015, preprint (arXiv:1505.05424)
Campbell H. et al., 2013, ApJ, 763, 88
Carrasco-Davis R. et al., 2018, PASP, 131, 108006
Charnock T., Moss A., 2017, ApJ, 837, L28
Chung J., Gulcehre C., Cho K., Bengio Y., 2014, preprint (arXiv:1412.3555)
Dai M., Kuhlmann S., Wang Y., Kovacs E., 2018, MNRAS, 477, 4142
D'Andrea C. B. et al., 2018, preprint (arXiv:1811.09565)
DeGroot M. H., Fienberg S. E., 1983, J. R. Stat. Soc. Ser. D (The Statistician), 32, 12
Foley R. J. et al., 2018, MNRAS, 475, 193
Fortunato M., Blundell C., Vinyals O., 2017, preprint (arXiv:1704.02798)
Frieman J. A. et al., 2008, AJ, 135, 338
Gal Y., Ghahramani Z., 2015a, preprint (arXiv:1512.05287)
Gal Y., Ghahramani Z., 2015b, preprint (arXiv:1506.02142)
Gieseke F. et al., 2017, MNRAS, 472, 3101
González-Gaitán S. et al., 2011, ApJ, 727, 107
Guo C., Pleiss G., Sun Y., Weinberger K. Q., 2017, preprint (arXiv:1706.04599)
Gupta R. R. et al., 2016, AJ, 152, 154
Guy J. et al., 2007, A&A, 466, 11
Hinton S. R. et al., 2018, ApJ, 876, 15
Hlozek R. et al., 2012, ApJ, 752, 79
Hochreiter S., Schmidhuber J., 1997, Neural Comput., 9, 1735
Hunter J. D., 2007, Comput. Sci. Eng., 9, 90
Ishida E. E. O., de Souza R. S., 2013, MNRAS, 430, 509
Ishida E. E. O., et al., 2018, MNRAS, 483, 2
Jones D. O. et al., 2018, ApJ, 857, 51
Kalchbrenner N. et al., 2018, preprint (arXiv:1802.08435)
Kendall A., Gal Y., 2017, preprint (arXiv:1703.04977)
Kessler R. et al., 2009, PASP, 121, 1028
Kessler R., Conley A., Jha S., Kuhlmann S., 2010a, preprint (arXiv:1001.5210)
Kessler R. et al., 2010b, PASP, 122, 1415
Kessler R. et al., 2018, MNRAS, 485, 1171
Kimura A., Takahashi I., Tanaka M., Yasuda N., Ueda N., Yoshida N., 2017, preprint (arXiv:1711.11526)
Kingma D. P., Ba J., 2014, preprint (arXiv:1412.6980)
Kuznetsova N. V., Connolly B. M., 2007, ApJ, 659, 530
Lanusse F., Ma Q., Li N., Collett T. E., Li C.-L., Ravanbakhsh S., Mandelbaum R., Póczos B., 2018, MNRAS, 473, 3895
LeCun Y., Cortes C., 2010, ATT Labs, 2
Lecun Y., Bottou L., Bengio Y., Haffner P., 1998, Proceedings of the IEEE, 86, 2278
Lochner M., McEwen J. D., Peiris H. V., Lahav O., Winter M. K., 2016, ApJS, 225, 31
LSST Science Collaboration, Abell P. A., Allison J., Anderson S. F., 2009, preprint (arXiv:0912.0201)
McKinney W., 2010, in van der Walt S., Millman J., eds, Proceedings of the 9th python science conference. p. 51
Mehri S., Kumar K., Gulrajani I., Kumar R., Jain S., Sotelo J., Courville A., Bengio Y., 2016, preprint (arXiv:1612.07837)
Möller A. et al., 2016, J. Cosmol. Astropart. Phys., 2016, 008
Moss A., 2018, preprint (arXiv:1810.06441)
Narayan G. et al., 2018, ApJS, 236, 9
Niculescu-Mizil A., Caruana R., 2005, Proceedings of the 22nd international conference on Machine learning (ICML '05). ACM, New York, NY, USA, p. 625
Pasquet J., Pasquet J., Chaumont M., Fouchez D., 2019, A&A, 627, A21
Paszke A. et al., 2017
Pedregosa F. et al., 2011, J. Mach. Learn. Res., 12, 2825
Poznanski D., Maoz D., Gal-Yam A., 2007, AJ, 134, 1285
Rubin D. et al., 2015, ApJ, 813, 137
Sako M. et al., 2011, ApJ, 738, 162
Smith L. N., 2018, preprint (arXiv:1803.09820)
Sutskever I., Vinyals O., Le Q. V., 2014, preprint (arXiv:1409.3215)
The Astropy Collaboration, 2018, AJ, 156, 123
The PLAsTiCC team et al., 2018, preprint (arXiv:1810.00001)
van der Walt S., Colbert S. C., Varoquaux G., 2011, Comput. Sci. Eng., 13, 22
Vasquez S., Lewis M., 2019, preprint (arXiv:1906.01083)
Waskom M. et al., 2018, Seaborn v0.9.0

APPENDIX A: REPRODUCIBILITY

To validate our code, we have independently reproduced the results of Gal & Ghahramani (2015a, b) and Fortunato et al. (2017) on a language modelling task and made the code public on GitHub.

APPENDIX B: BEHAVIOUR OF OOD IN MNIST

To better understand the behaviour of Bayesian neural networks, we carried out two experiments on the MNIST data set (LeCun & Cortes 2010). This data set contains images of handwritten digits.

First, we trained three (standard, MC dropout, and BBB, respectively) feed-forward networks to classify flattened digit images into one of ten classes, each representing a digit between 0 and 9. To evaluate performance on OOD data, we asked the network to carry out predictions on rotated digits, as well as completely randomized images. In Fig. B1, we show the predictions for the standard feed-forward network where we see it remains confident about its prediction (the maximum of the predicted probability remains relatively high) for rotated images as well as random images. For MC dropout and BBB networks, we observe a much more interesting behaviour (Figs B2 and B3). When the rotation is

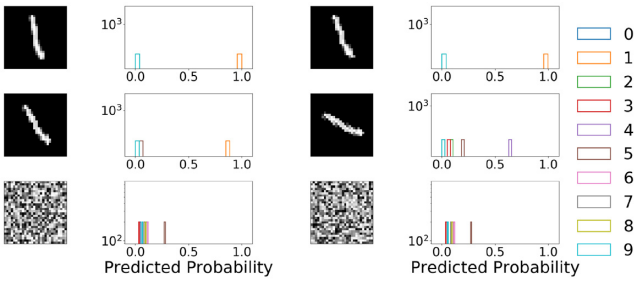

Figure B1. A standard feed-forward network on MNIST. On the first and third columns, images show the input given to the network. The classification probabilities for each target are given as histograms in the second and third columns.

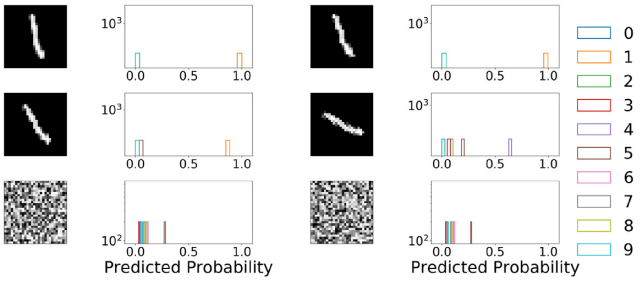

Figure B2. An MC Dropout feed-forward network on MNIST.

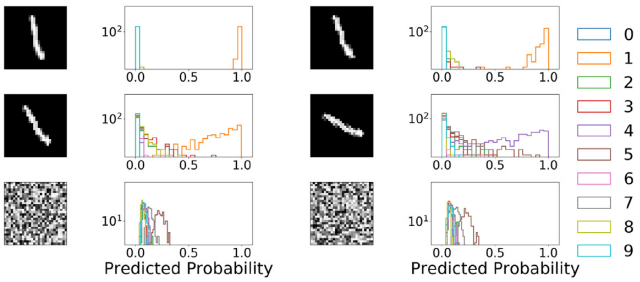

Figure B3. A Bayesian by Backprop feed-forward network on MNIST.

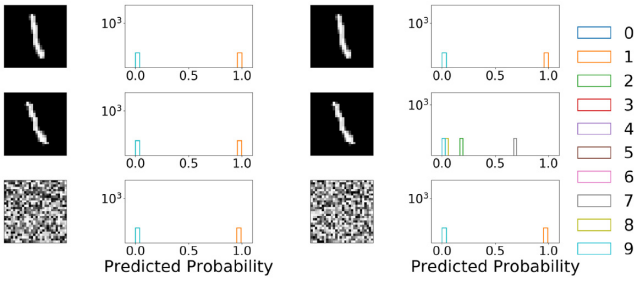

Figure B4. A recurrent network on MNIST. This RNN is able to obtain similar prediction behaviour as Fig. B1, which is what is expected for OOD events.

small, the network behaves in much the same way as a non-Bayesian one and gives maximum probability with very little variance to the correct class. However, as the rotation increases, the variance of the prediction increases significantly. Clearly, then the network is uncertain about the correct class, but the fact that it still sometimes assigns a high score to one of the classes indicates that it believes that the image bears some similarity to the data it was trained on. This behaviour is in contrast with that on random images: The predicted probabilities are all concentrated to the left, with very little variance: The network is confident that it does not know how to predict this image. This is the expected behaviour for OOD samples, which yields a large entropy.

Secondly, we trained three RNNs (standard, MC dropout, and BBB, respectively), similar to the ones used in SuperNNova, to carry out the same task. The digit images were treated as a sequence, with the rows as the time-steps and the columns as input features. The standard network behaves in much the same way as its feed-forward counterpart (Fig. B4) and makes even more confident predictions on the random and rotated images. For Bayesian recurrent networks, we found that it was harder to obtain desirable OOD behaviour as

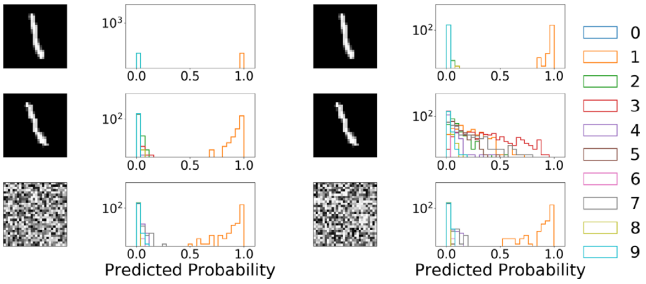

Figure B5. An MC Dropout recurrent network on MNIST. This network collapses and outputs high-certainty predictions for OOD images.

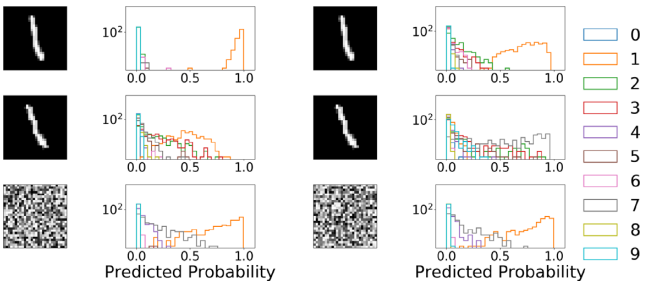

Figure B6. A Bayesian by Backprop recurrent network on MNIST. This network collapses and outputs high-certainty predictions for OOD images.

exhibited by the Bayesian feed-forward networks, and that extensive parameter tuning was required. The MC dropout and BBB networks (Figs B5 and B6) show, as before, an increased variance in the prediction probabilities for random images. However, the behaviour on random data differs significantly. The MC dropout network seems to have collapsed on predicting a single class with high probability while the Bayesian network exhibits a large variance for multiple classes.

While we have verified that tuning the various hyper-parameters improves the uncertainty performance on this qualitative examination, it is clear that the behaviour of Bayesian recurrent networks should be critically analysed: The network remains at risk to collapse its predictions when fed with unrelated data. This sheds light on the negative ΔH found in Section 5.3: For OOD data, which looks nothing like the training data, the network likely collapses and outputs a prediction with very high certainty, giving a very low entropy score to the out-of-sample data. We note that this is possibly exacerbated by the type of data used to train the network: Supernova fluxes indeed exhibit variations spanning multiple orders of magnitude, which leads to persisting artefacts even after normalization. Future work will focus on characterizing this phenomenon and developing methods to improve robustness on OOD data.

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