SuperNNova: an open-source framework for Bayesian, Neural Network based supernova classification

A. Möller,[1,2]* T. de Boissière[3] †

[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] *Lyrebird AI, Unit 302, 55 Mont-Royal Avenue Ouest, Montreal, QC H2T 2S5, Canada.*

ABSTRACT

We introduce SuperNNova, an open source supernova photometric classification framework which leverages recent advances in Deep Neural Networks. Our core algorithm is a recurrent neural network (RNN) that is trained to classify light-curves using photometric information only. Additional information such as host-galaxy redshift can be incorporated to improve performance. We evaluate our framework using realistic supernovae simulations that include survey detection. We show that our method, for the type Ia vs. non Ia supernovae classification problem, reaches accuracies greater than 96.51 ± 0.12 without any redshift information and up to 99.51 ± 0.08 when redshift, either photometric or spectroscopic, is available. Further, we show that our method attains unprecedented performance for classification of incomplete light-curves, reaching accuracies > 84.02±0.21 (> 92.21±0.81) without host-galaxy redshift (with redshift information) two days 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 datasets 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 supernovae 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 on GitHub ⬡ 📘.

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 spectroscopic follow-up of promising supernova candidates identified by photometric surveys. Previous and current surveys such as SNLS, SDSS, the Foundation Supernova Survey and Dark Energy Survey have discovered thousands of supernovae (Astier et al. 2006; Frieman et al. 2008; Foley et al. 2018; Bernstein et al. 2012). 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 et al. 2009). This

survey, together with others such as the Zwicky Transient Survey (ZTF), will explore transient events and expect to have up to a million transient alerts per night (Bellm 2018). The scarcity of spectroscopic resources is, and will continue to be, a limiting factor for supernovae 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 color 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 et al. 2007; Sako et al. 2011) or classification using features extracted from 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; 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 supernovae 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) dataset 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 and many are not suited for real-time classification (Poznanski et al. 2007; Sako et al. 2011; Ishida, E. E. O. and de Souza, R. S. 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; Jones et al. 2018; Hinton 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 2015b,a; Fortunato et al. 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 supernovae 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 which 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 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 which will be a major focus of brokers such as LSST's 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 not biased by template matching nor limited by costly feature engineering. We obtain state-of-the-art results on the classification of supernovae with complete and partial light-curve data. We show that SuperNNova's performance increases with the amount of data and show that it readily scales to very large datasets. The performance is further enhanced with additional information such as host-galaxy redshifts. Crucially, we show that 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 out-of-distribution 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 analyze the effect of a photometrically classified type Ia SN sample in cosmology.

2 METHODS

2.1 Simulations and available datasets

We simulated realistic supernovae light-curves using the SNANA package (Kessler et al. 2009). Our data is similar to the SPCC data (Kessler et al. 2010a; Kessler et al. 2010b) with updated models used in the DES simulations (Kessler et al. 2018). DES light-curves are built from supernova templates and use SALT2 SN Ia SED models (Guy et al. 2007) and the trained model from JLA (Betoule et al. 2014). Observing logs specifying the simulated cadence and conditions were included when available.

Our simulation contains $1,983,213$ supernovae light-curves of which $881,969$ are successfully fitted using SALT2 to obtain features such as color and stretch. SALT2 fits do not always converge and therefore some light-curves can not be fitted. Furthermore, we require loose cuts for fitting, such as: an observation with $S/N > 5$, and at least 1 epoch before -2 days, at least 1 epoch past $+30$ days (rest-frame). 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 sub-sample

Table 1. Simulated supernovae samples divided by subtypes. We show the number of supernovae light-curves available per type in the complete dataset and the number that have a succesful 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 dataset
Ia	402,786	912,691
Ib	140,197	181,454
Ic	70,811	90,485
IIP	94,994	296,523
IIn	3,249	154,614
IIL	93,535	189,615
	IILs by template	
IIL1	26,717	100,827
IIL2	66,818	88,788

the dataset to make sure classes are balanced. For binary classification i.e. the discrimination of SNe Ia vs. 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% training, 10% validation and 10% testing.

In the following we report our results separately for the *"complete"* and the *"SALT2 fitted"* datasets. 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"* dataset a non-representative one, as it fails to explore the full diversity of supernovae. Our representative set will be the *"complete"* dataset. We further explore these samples and the issue of representativeness in Section 5.2.

2.2 Evaluation metrics

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.

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:

$$\text{accuracy} = \frac{\text{TP} + \text{TN}}{\text{TP} + \text{TN} + \text{FP} + \text{FN}} \tag{1}$$

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:

$$\text{purity} = \frac{\text{TP}}{\text{TP} + \text{FP}}; \quad \text{efficiency} = \frac{\text{TP}}{\text{TP} + \text{FN}} \tag{2}$$

Contamination by core collapse is defined as:

$$\text{contamination} = \frac{\text{FP}}{\text{FP} + \text{TN}} \tag{3}$$

In this work, performance metrics reported with errors represent the mean and one standard deviation of the distribution. This distribution is obtained by performing 5 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 modeling sequential data or time series data (Mehri et al. 2016; Bahdanau et al. 2014). 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 2 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 parameterized by learnt weights $(W, W', V, U, U'$ in Figure 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 bidirectional: it is allowed to process information from left to right as well as 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. Gal et al. (Gal
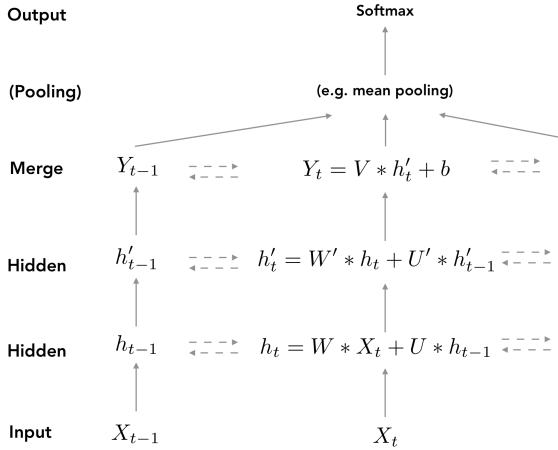
Figure 1. Partial schema of a vanilla bidirectional RNN with mean pooling. Layers are indicated in the first left column. Inputs (X_t) are given to the network. Arrows indicate transmission of temporal information. The network extends to the right by the number of $t \in [1, T]$ available inputs.

& Ghahramani 2015b) show that dropout (a regularization technique consisting in randomly dropping weights or activations) has a Bayesian interpretation and can be used to improve performance and calibrate predictions. Fortunato et al. (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 modeling 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 ⬤ with accompanying documentation 📄.

The core building block of our algorithm is a Recurrent Neural Network (RNN). Our RNN processes the photometric time series to produce a sequence of hidden states. The sequence is condensed to fixed length through mean pooling or alternatively by concatenating the last hidden state of each recurrent layer. Finally, a linear projection layer is applied to obtain an 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 hyperparameters 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 "Variational" implements (Gal & Ghahramani 2015b) and the second one, which we simply call "BBB" implements (Fortunato et al. 2017). 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, seaborn (McKinney 2010; Van Der Walt et al. 2011; The Astropy Collaboration et al. 2018; Astropy Collaboration et al. 2013; Hunter 2007; Paszke et al. 2017; Pedregosa et al. 2011; Waskom et al. 2018).

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. 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.

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

RNN training details

The light-curve data is typically formatted in a tabular way, where each row corresponds to one observation, in one specific filter. We pivot the table such that observations within eight hours of each other are grouped 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_g, FLUXCAL_i, FLUXCAL_r, FLUXCAL_z), flux errors (FLUXCALERR_g, FLUXCALERR_i, FLUXCALERR_r, FLUXCALERR_z), time step (delta_time) and available flux information (g, gi, gir, girz, giz, gr, grz, gz, i, ir, irz, iz, r, rz, 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 center 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 which decays the learning rate by a constant factor when the validation

loss stagnates and the 1-cycle policy introduced in (Paleari et al. 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 dataset 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, we batch together multiple light-curves at a time, using zero-padding and masking where appropriate.

2.5 Random Forest (RF)

To compare our Recurrent Neural Networks with other classification methods, we implement an additional classifier. We use the excellent scikit-learn (Pedregosa et al. 2011) library to implement a random forest classifier, which has been previously shown to have high performance in supernova photometric classification tasks (Möller et al. 2016; Lochner 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 random forest classifier, the information implicitly percolates through the SALT2 fit parameters. We retained the following features to train our random forest classifier: stretch $(x1, x1_{ERR})$, color (c, c_{ERR}), magnitude in B-band (mB, mB_{ERR}), normalization $(x_0, x_{0_{ERR}})$, goodness of fit (χ^2), and observed magnitudes in $rigz$ bands $(m_{0obs_r}, m_{0obs_i}, m_{0obs_g}, m_{0obs_z}$ and their errors).

3 BASELINE RNN

We introduce a "Baseline RNN" which 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 Figure 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* dataset. 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), 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
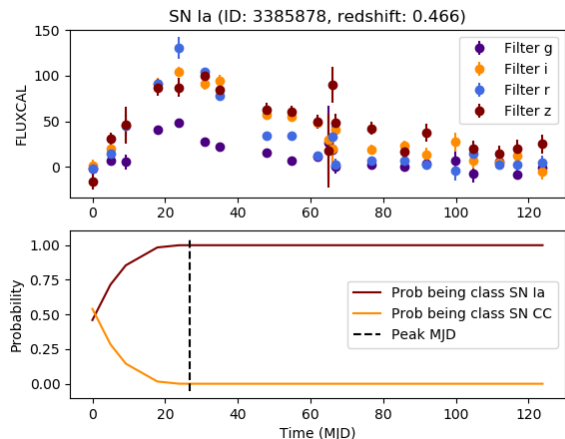
Figure 2. Simulated type Ia supernova light-curve and classification. Top: calibrated flux evolution in different DES band-passes as a function of normalized time (the first photometric measurement is set to time equals zero). Bottom: Baseline RNN classification probability evolution with respect of 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 gray dashed line and simulated redshift of the supernovae is shown on the top $z = 0.466$ 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%). 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 common practice to facilitate the training of neural networks. With half of the *SALT2 fitted* dataset, 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 (95.6 ± 0.8) when compared to no normalization (94.3 ± 0.5). 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 (95.9 ± 0.4) to global normalization. Such a normalization scheme blurs the color

ratio between observed light-curves in different band-passes. Interestingly, this indicates that color 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 the Random Forest introduced in Section 2.5. We use the same training and validation samples from the *SALT2 fitted* dataset to evaluate both algorithms. We assess uncertainties in the accuracy by performing 5 randomized runs.

Using the whole *SALT2 fitted* dataset, both the baseline RNN and RF are found to have high-accuracy 96.5 ± 0.1 and 95.14 ± 0.07 respectively and an AUC of 0.9956 ± 0.0002 and 0.9927 ± 0.0003 each without any redshift information. Our Baseline RNN has a higher accuracy and AUC than many methods found in the literature (Möller et al. 2016; Lochner et al. 2016; Dai et al. 2018; Charnock & Moss 2017; Moss 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 Figure 3, we find that our Baseline RNN outperforms the Random Forest for large dataset sizes given the same redshift information. We also find that RNNs exhibit an increase in performance with larger datasets while the Random Forest classifiers remains almost constant. Again, we stress that even when no explicit redshift information is provided to the Random Forest 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).

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) uses half of the PCC dataset 5 times data-augmented, equivalent to $\approx 50,000$ supernovae light-curves, and obtains 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 dataset, we can train our baseline RNN with $32,222$ light-curves and obtain a classification accuracy of 92.47 ± 0.8 and an AUC of 0.9805 ± 0.0038 without redshift information, when using photometric redshifts we obtain an accuracy of 96.56 ± 0.53. 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% 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% for photometric and 3% 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 in the order of a couple percent for Ib and Ic SNe and under 1% for other types. Contamination levels are found to be not correlated with the number of light-curves for training. Within this 1% 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 2.37 ± 0.13 and 1.29 ± 0.09 respectively to $< 0.1\%$ with host redshifts. Interestingly, 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 (Kessler et al. 2018; D'Andrea et al. 2018). Our Baseline RNN performs superbly, with almost constant efficiency as a function of simulated redshift with an average efficiency of 97.41 ± 0.16 without and up to 99.54 ± 0.09 with host-galaxy redshift information. Such high efficiencies enable probing new supernovae populations at high-redshift 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 limitation 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 compromises the purity of the sample. On the contrary, the purity of a SN Ia sample classified without redshift is 95.67 ± 0.27, while the addition of redshifts increases the sample purity to 99.48 ± 0.08. This level of contamination, less than 1%, is within the current range of contamination of spectroscopically classified samples currently used in cosmological and astrophysical analyses (Rubin et al. 2015).
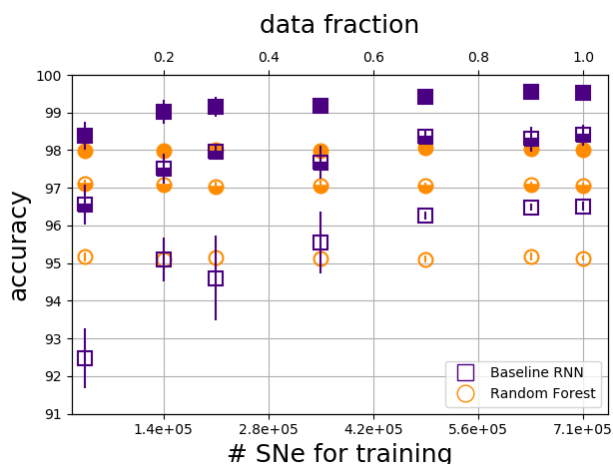
Figure 3. Ia vs. non-Ia classification accuracy with respect to training set size using the *SALT2 fitted* dataset. Bottom axis indicates the used data fraction and the top axis number of supernovae. Accuracy of baseline RNN is shown in squares and Random Forest in circles. Error bars are one standard deviation from the accuracy distribution for five runs with different random seeds. Classification using: 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 random forest for the same dataset and explicit redshift information.

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* dataset which 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* datasets. Additionally, since training set numbers affect performance metrics, we compare the performance with 43% of the *complete* dataset. This fraction of the complete dataset has a similar size training set than the *SALT2 fitted* one.

For all datasets, our baseline RNN is capable of attaining a high-accuracy classification, > 84% right before the maximum of a supernova light-curve. Further, we show that the addition of host-galaxy redshifts produce 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*

Table 2. Ia vs. non-Ia classification accuracy using baseline RNN trained with whole *SALT2 fitted* dataset, a fraction of 0.43 and the whole *complete* dataset. Accuracy for partial light-curve classification with respect of 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.

	SALT2 fitted dataset			
redshift	-2	0	+2	all
None	84.02 ± 0.21	85.36 ± 0.21	86.72 ± 0.19	96.51 ± 0.12
zpho	93.26 ± 0.44	93.88 ± 0.5	94.5 ± 0.54	98.4 ± 0.28
zspe	96.92 ± 0.26	97.36 ± 0.28	97.75 ± 0.25	**99.51 ± 0.08**

	43% of complete dataset			
redshift	-2	0	+2	all
None	85.94 ± 0.15	87.08 ± 0.14	88.15 ± 0.14	96.63 ± 0.05
zpho	93.2 ± 0.15	93.87 ± 0.13	94.47 ± 0.12	**98.63 ± 0.06**
zspe	92.21 ± 0.81	92.96 ± 0.77	93.51 ± 0.77	97.84 ± 0.45

	Complete dataset			
redshift	-2	0	+2	all
None	86.47 ± 0.16	87.59 ± 0.13	88.68 ± 0.11	96.97 ± 0.06
zpho	93.56 ± 0.06	94.25 ± 0.07	94.84 ± 0.06	**98.83 ± 0.02**
zspe	93.36 ± 0.15	94.09 ± 0.14	94.66 ± 0.14	98.43 ± 0.07

fitted sample and it may be explained by a selection bias in the *complete* dataset 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 9.17 ± 0.19% contribution and Ib with 3.27 ± 0.07%. Interestingly, type IIP and IIL2 supernovae can contribute around 2% 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 84.02 ± 0.21 up to 96.92 ± 0.26 for the *salt fitted* dataset, without and with redshifts respectively. SuperNNova opens a path towards efficient use of photometric and spectroscopic resources for 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 supernovae types

There is more to supernovae classification than binary classification. Time-domain surveys are increasingly exploring the diversity of supernovae and would benefit from classification of multiple supernova classes. We explore ternary (Ia, Ibc and IIs) and seven-way (Ia, IIP, IIn, IIL1, IIL2, Ib, Ic)

Table 3. Ternary (Ia, Ibc, IIs) and seven-way (Ia, IIP, IIn, IIL1, IIL2, Ib, Ic) classification using baseline RNN trained with complete dataset. Accuracy for partial light-curve classification with respect of 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.

Ternary classification

redshift	-2	0	+2	all
None	69.36 ± 0.34	71.76 ± 0.33	74.39 ± 0.3	92.64 ± 0.16
zpho	79.62 ± 0.25	81.44 ± 0.32	83.35 ± 0.32	$\mathbf{95.4 \pm 0.15}$
zspe	78.83 ± 0.18	80.7 ± 0.17	82.59 ± 0.17	94.99 ± 0.12

Seven-way classification

redshift	-2	0	+2	all
None	57.2 ± 0.31	60.08 ± 0.34	62.99 ± 0.32	86.89 ± 0.2
zpho	64.69 ± 0.21	67.32 ± 0.26	69.96 ± 0.25	90.02 ± 0.14
zspe	63.99 ± 0.58	66.74 ± 0.62	69.43 ± 0.65	$\mathbf{90.14 \pm 0.47}$

classification tasks. We train with the *complete* dataset to obtain a large number of light-curves per target.

For 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 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. Nonetheless, 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 vs. non Ia) as shown in Figure 3. Splitting the core-collapse supernovae in subclasses adds a new level of complexity which accounts for the performance drop. Interestingly, for seven-way classification the contamination of the SN Ia sample is dominated by type IIP SN as seen in Figure 4. This was seen for early classification in previous Section 3.5 but not for the complete light-curve classification.

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 an interesting avenue for further studies. It is possible that a two-step procedure is useful, where: first a multi-target 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.
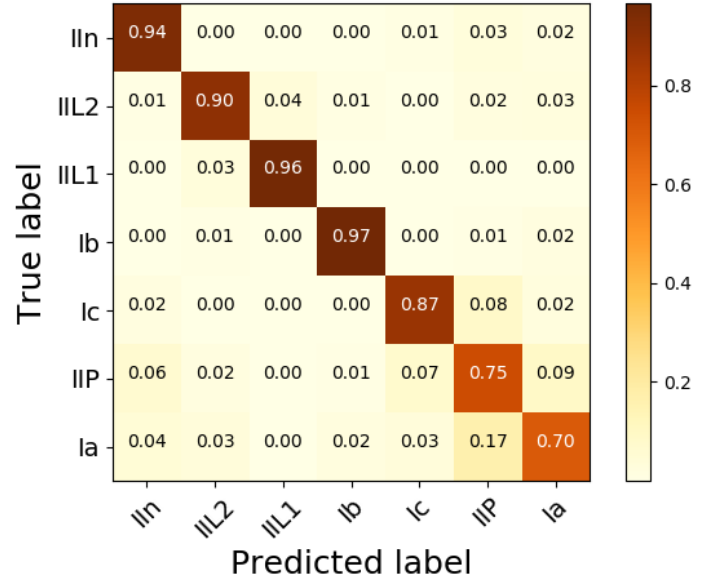

Figure 4. Confusion Matrix for seven-way classification with best performing Baseline classifier with no redshift information. Matrix shows the fraction of light-curves for a given supernova type (True label) classified as the supernova type in the horizontal axis (Predicted label). The diagonal elements are those light-curves correctly classified while off-diagonal elements are those that are mislabeled by the classifier. Color bar indicates the normalized percentage of a certain type of SN light-curves in the predicted label. For seven-way classification our algorithm performs superbly for types Ib, IIL1 and IIn SNe while SNe Ia are not as well characterized.

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 labeled training observations $\mathcal{D} = (x_i, y_i)_{i=1,\ldots N}$, we minimize the negative log likelihood $NLL = \min_{\mathbf{w}} \sum_{i=1}^{N} -\log \mathcal{P}(\mathbf{y_i}|\mathbf{x_i}, \mathbf{w})$ with gradient descent. Weights are determined during training and then used to classify $\hat{\mathbf{y}}$ new, unlabeled data.

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, unlabeled data points: $\mathcal{P}(\hat{\mathbf{y}}|\mathbf{x}) = \int \mathcal{P}(\hat{\mathbf{y}}|\mathbf{x}, \mathbf{w}) \mathcal{P}(\mathbf{w}|\mathcal{D}) \, d\mathbf{w}$. 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 standard deviation of the distribution. Neural networks are then trained to minimize a divergence between the *posterior distribution* and its *variational approximation*:

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

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

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

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 which 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 dataset for training, validation and testing.

4.2 Variational

Following (Gal & Ghahramani 2015b,a) we define our variational distribution to factorize over each row of the network's weight matrices: $q(\mathbf{w}_k) = p\mathcal{N}(0, \sigma^2 I) + (1-p)\mathcal{N}(\mu_{\mathbf{k}}, \sigma^2 I)$ where p is the dropout probability, $\mathcal{N}$ the normal distribution, μ_k is the variational parameter we optimize over with our gradient descent algorithm, σ^2 is a fixed, small constant and I 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 2015b,a).

We can now obtain a distribution of predictions, simply by sampling a different dropout mask for each prediction. In this work we sample predictions fifty times for each light-curve. The median of this array is used to report the accuracy score. These uncertainties can provide valuable information on 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, $[0.01, 0.05, 0.1, 0.2]$ and weight decay $[1e^{-5}, 1e^{-7}, 1e^{-9}]$ 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%, we choose to use weight decay $1e^{-7}$ in the following. For the *complete* dataset this configuration has an accuracy of 96.83 ± 0.11 without redshift and 98.79 ± 0.04 with host photometric redshift for complete light-curve classification which is equivalent to the baseline RNN accuracy with a

sligth reduction. We find that the efficiency (98.95 ± 0.07), purity (98.63 ± 0.05) 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 Out-of-Distribution events.

4.3 Bayes by Backprop (BBB)

Following Fortunato et al. (2017), we use a simple Gaussian distribution to model the variational distribution.

During training, we sample each weight as follows: sample $\epsilon_i \sim \mathcal{N}(0, 1)$ then obtain $w_i = \mu_i + \sigma_i \times \epsilon_i$. Where σ_i, ϵ_i are optimized by gradient descent. We compute the loss of a minibatch of samples as:

$$\frac{1}{B}\left[\sum_{i=1}^{B} -\log(\mathcal{P}(y_b|\theta, x_b))\right] + \mathbf{KL}(q(\mathbf{w}|\theta))||\mathcal{P}(\mathbf{w})) \tag{7}$$

where the KL-divergence is estimated with Monte Carlo sampling and the prior is given by $\mathcal{P}(\mathbf{w}) = \pi\mathcal{N}(0, \sigma_1) + (1 - \pi)\mathcal{N}(0, \sigma_2)$.

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 saltfit dataset. 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 $\log\left(e^{\frac{\sigma_{\text{mix}}}{\rho_{\text{lower}}}} - 1\right)$ and $\log\left(e^{\frac{\sigma_{\text{mix}}}{\rho_{\text{upper}}}} - 1\right)$, where $\sigma_{\text{mix}} = \sqrt{\pi\sigma_1^2 + (1 - \pi)\sigma_2^2}$, $\rho_{\text{lower}} \in \{4, 3\}$ and $\rho_{\text{lower}} \in \{2, 1\}$ for the recurrent layer and $\rho_{\text{lower}} \in \{3, 2, 1\}$ and $\rho_{\text{lower}} \in \{2, 1\}$ for the output layer.

We found that the Bayesian network trains and attains better performance without the cyclic learning rate. This may be due to the larger number of variables in this implementation. The most accurate model has a $\log \sigma_1 = -1$, $\log \sigma_2 = -7$, $\rho_{\text{lower}} = 4$ and $\rho_{\text{upper}} = 3$ for the recurrent layer and $\log \sigma_1 = -0.5$, $\log \sigma_2 = -0.1$, $\rho_{\text{lower}} = 3.0$ and $\rho_{\text{upper}} = 2.0$ for the output layer. This model reaches an accuracy of 96.76 ± 0.02 without redshift and 98.69 ± 0.03 with host photometric redshift information for whole light-curve classification in the *complete* dataset. The efficiency, purity and contamination values are similar to those of the Variational implementation and slightly reduced with respect to the Baseline implementation $\approx 0.5\%$. In Section 5, we will discuss other criteria for selecting the best performing method, through calibration, entropy of Out-of-Distribution events and model uncertainties.

4.4 BRNNs 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 dataset (through simulated flux measurement errors). By casting learning in a Bayesian framework, we expect to model *epistemic* uncertainties.

For BRNNs, given a light-curve X_k with a classification probability $p(X_k)$, the uncertainty of the prediction is defined as:

$$\mu_k = std\left(\sum_{j=1}^{N} p_j(X_k)\right) \tag{8}$$

where $j \in [1, N]$ is the number of inference samples 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:

• *Mean prediction uncertainty*: for a given test set, we average the prediction uncertainties (Eq. 8) over all light-curves:

$$\mu = \frac{1}{K}\sum_{k=1}^{K}\mu_k. \tag{9}$$

For two given set of predictions, we can compute the difference between their uncertainties as:

$$\delta\mu_s = \mu_1 - \mu_2. \tag{10}$$

• *Entropy*: we follow Fortunato et al. (2017) and define, for a given input sequence $x = X_{t \in [1,T]}$ and a classification model m, the entropy of x under m as:

$$H_m[x] = \sum_w p(c|x)log\frac{1}{p(c|x)}. \tag{11}$$

where $p(c|x)$ is the classification probability given the input sequence. 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 $\overline{H_m}[x] = \frac{1}{N}H_m[x]$, where N is the number of light-curves in a given test set. For two given set of predictions, we can define the *entropy gap* ΔH_m by:

$$\Delta H = \quad \overline{H_{m1}}[x] - \overline{H_{m2}}[x] \quad or \quad = \overline{H_m}[x_1] - \overline{H_m}[x_2]. \tag{12}$$

where the first option evaluates the entropy gap over the same dataset for two given models, and the second uses the same model to make predictions for two different data sets.

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 which represents the set of plausible models, given
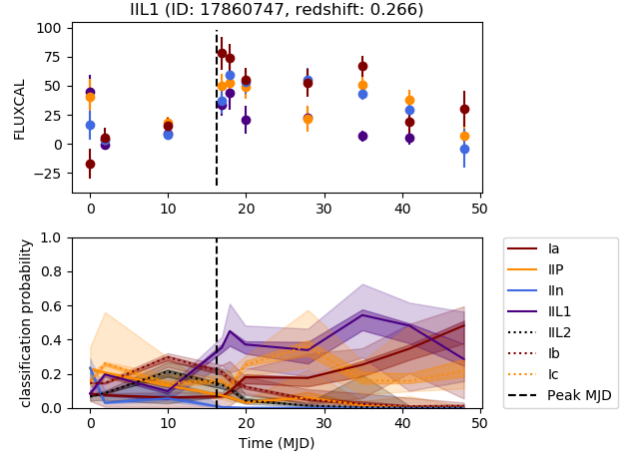

Figure 5. Classification of simulated type IIL1 SN as type Ia SN with seven-way Variational RNN. The model was trained without redshift information. Predictions for this light-curve have low confidence and have large uncertainties (shadowed regions show 68% and 95% contours). The large uncertainties on the classification probability are representing the lack of confidence in this classification and are correlated to the low S/N.

the data. As more data is 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 $\delta\mu$ (Eq. 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* dataset we compute the $\delta\mu$ of models trained with 43% and the whole training set, $\delta\mu = \mu_{0.43} - \mu_{whole}$. We find $\delta\mu = 0.004$ and 0.007 respectively. We find similar values for the $\delta\mu$ of models trained with half and the *salt fitted* training set.

Second, we compute the entropy gap defined in Eq. 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 dataset, we should obtain a positive ΔH. We find for the Variational and BBB implementations a positive $\Delta H > 0.01$.

Both BRNNs implementations have uncertainties which are consistent with the behavior expected of *epistemic* uncertainties as shown by the metrics computed above. However, we find that the size of uncertainties differ in both methods. If we compute the mean μ_s when classifying the *complete* dataset, we find that the variational 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 behavior of our BRNNs uncertainties. In particular, we will explore the effect of non-representative training sets and the classification of out-of-distribution 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 vs. non Ia classification task.

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 Random Forest, 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 analyze our algorithms' calibration, we use reliability diagrams (DeGroot & Fienberg 1983). These diagrams are constructed by discretizing the predicted probability into ten 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 probability 0.7, we expect 70% of events to be true SNe Ia and 30% to be missclassified SNe from other types. Furthermore, we construct a metric to study the calibration deviation: the difference between the two calibrations squared.

For the Random Forest algorithm, we find a large calibration deviation when classifying the *SALT2 fitted dataset*, as can be seen in Figure 6. Over five randomized runs the level of dispersion is found to be 0.0242 ± 0.0008. In this classification task, RNNs are found to have better calibration than the Random Forest algorithm with a dispersion an order of magnitude lower. For BBB and Variational RNNs we construct 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 uncertainties.

Bayesian RNNs are found to be better calibrated than the Baseline RNN for both *salt fitted* and *complete* datasets. For the classification of the *complete* dataset without any redshift information, we find a calibration dispersion from the Baseline RNN of 0.005 ± 0.001 which is reduced to 0.003 ± 0.002 for the Variational and to 0.002 ± 0.001 for the BBB implementations.
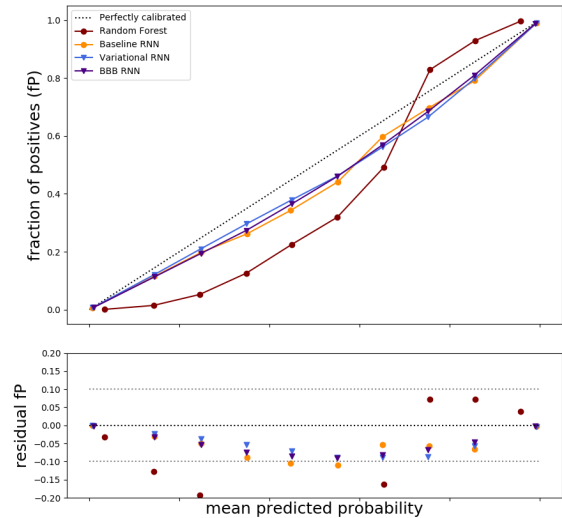

Figure 6. Calibration of classification algorithms. Top: reliability diagram showing the calibration for *SALT2 fitted* dataset classification for a single seed. We use the most accurate configurations for the Random Forest (red circles), Baseline RNN (purple triangles), Variational RNN (yellow circles) and BBB RNNs (blue triangles). Bottom: dispersion from perfectly calibrated algorithms. Note that the Random Forest algorithm has a large deviation from perfect calibration while the RNNs are better calibrated than this algorithm with the BB implementation almost perfectly calibrated.

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 dataset* 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* dataset and 0.43 of the *complete* dataset. We find that the data fraction or nature of the dataset can change the dispersion up to 50%.

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 datasets where large calibration dispersion is observed. We consider our BBB and Variational RNNs to be well calibrated due to deviations less than one percent.

5.2 Representativeness

To improve generalization, classification algorithms typically require large and representative training sets. Current and past surveys have spectroscopic follow-up strategies designed to target as many Ia-like objects as possible. Spectroscopically selected samples are therefore not probing supernovae diversity. Furthermore, these samples are then used to generate simulated SN light-curves, thus biasing our photometric classifiers. In this section, we investigate the impact of training SuperNNova with non-representative and representative datasets.

Previous works have discussed the issue of representativeness by comparing the accuracy of an algorithm trained with a dataset whose size was similar to the test set (Lochner et al. 2016; Charnock & Moss 2017). RNN based methods, trained on small datasets, are prone to over-fitting (Charnock & Moss 2017). Additionally, we have shown that the performance of our algorithm depends on the size of the training set (see Figure 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* dataset as an example of non-representative data. Recall that this dataset 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* dataset which 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 datasets is shown in Figure 7. To investigate the discrepancy between spectroscopic and photometric samples, we train with the non-representative dataset and evaluate the classification performance for the test sample in the representative dataset.

For classification with no redshifts using the Baseline and Variational RNNs we find that the accuracy is reduced by 0.3% when trained with a non-representative dataset. 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 non-representative set (in our case, the full *SALT2 fitted* dataset) to be more uncertain than one trained on a representative set (in our case, a subset of the *complete* dataset) 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 metrics to be positive for all BRNNs, $\delta\mu > 0.005$ 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 datasets. Bayesian RNN uncertainties may be a promising indicator to select follow-up candidates for this purpose. This is an interesting possibility which we leave for future work.
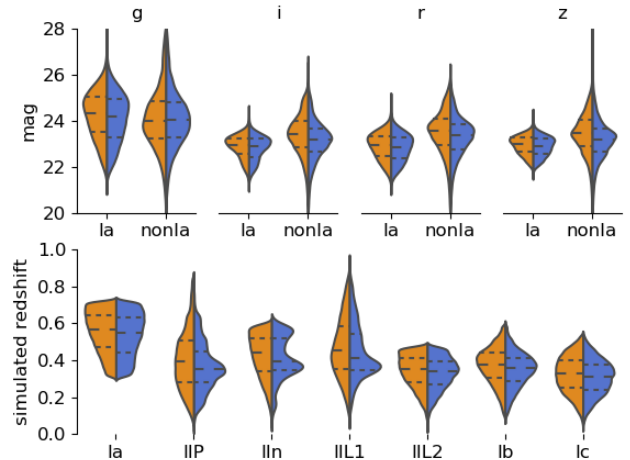

Figure 7. Distributions of maximum observed brightness (mag), in all DES filters (g, i, r, z), and of simulated redshift for *SALT2 fitted* (left yellow) and *complete* (right blue) datasets. Maximum observed brightness is shown for type Ia and non-Ia samples while simulated redshift is shown for each of the available templates. Dashed lines show the median and first quartile of the distribution. The *complete* and *SALT2 fitted* datasets probe similar parameter space but have different distributions. This is similar to what is expected of non-representative samples.

5.3 Out of distribution light-curves (OOD)

In astronomy, as in any other classification application, the generalization properties of a classifier and its behavior on unseen, possibly out-of-distribution samples represents a challenge. In this section we study the performance of SuperNNova when classifying out-of-distribution 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 out-of-distribution light-curves, all SuperNNova algorithms rarely classify these light-curves as SNe Ia. For binary classification, the reverse and shuffle light-curves obtain the largest number of classifications as SNe Ia, 6.2% and 3.9% respectively with the variational implementation and less than 3% for the BBB. Many of these light-curves resemble supernovae, specially with light-curves with low signal-to-noise. For the sinusoidal and random light-curves less than 3% are classified as SNe Ia in the variational implementation and < 3.94% for Baseline and BBB implementations. The network seems to characterize type Ia supernovae well and therefore classifies most OOD events as core-collapse supernovae as can be seen in Figure 8. In ternary and seven-way classification the most common predictions for OOD events are types II: IIn, IIp, IIL1.

To assess our model probability distributions with respect to out-of-distribution events, we compute ΔH for our best performing models comparing OOD and our *complete* data test set predictions. For the binary classification prob-
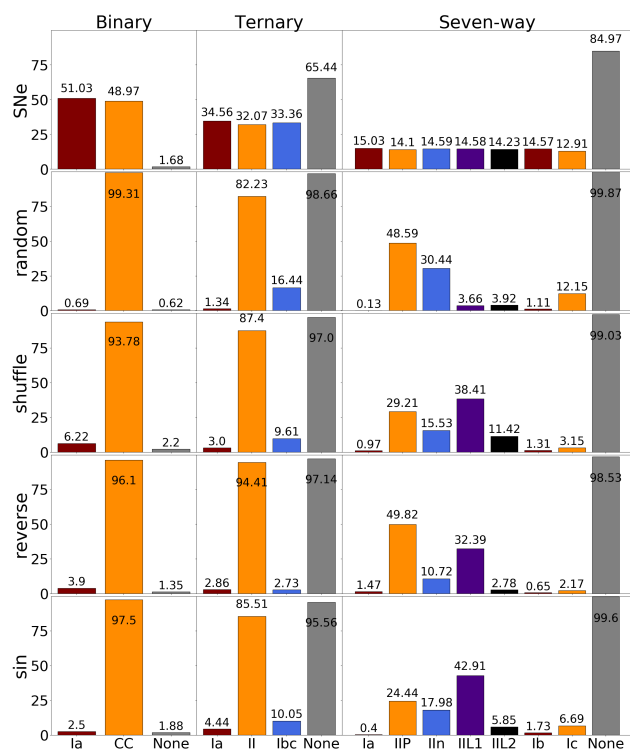
Figure 8. Predicted classification percentages for best configuration variational RNN. Columns indicate number of targets for the classification (with probabilities summing one) and an additional column indicating a low probability prediction, $< 0.6, 0.4, 0.2$ for 2, 3 and 7 classes respectively. Rows represent the different light-curve types. All rows are out-of-distribution generated light-curves except the row "SNe" representing our testing dataset. Since the testing set has balanced classes, we expect the prediction percentage in the row "SNe" to be balanced as well. Note that the out-of-distribution events are rarely classified as type Ia SNe ($< 6.22\%$ binary, $< 4\%$ ternary and $< 1.5\%$ in seven-way classification), the highest percentages in binary classification are for OOD which can resemble SNe light-curves, e.g. reverse and shuffle. In ternary and seven-way classification, out-of-distribution events are preferably classified as type II, IIn or IIL1 SNe. Note that the number of low probability detections are much higher for OOD when classifying in three or seven classes.

lem, we find positive ΔH for random and reverse light-curves for both the Variational and BBB implementations, with the BBB having the largest entropy gap. Interestingly, for sinusoidal and shuffle light-curves this metric is negative for the Variational implementation. For the ternary and seven-way problems, other OOD predictions are as well negative. This is unexpected behaviour which we explore in Appendix 7 where we conclude that RNNs are at risk of collapse on predicting a single class with high probability. Additionally, we show the difference behaviours for classification of OOD events in a seven-way classification in Figure 9.

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
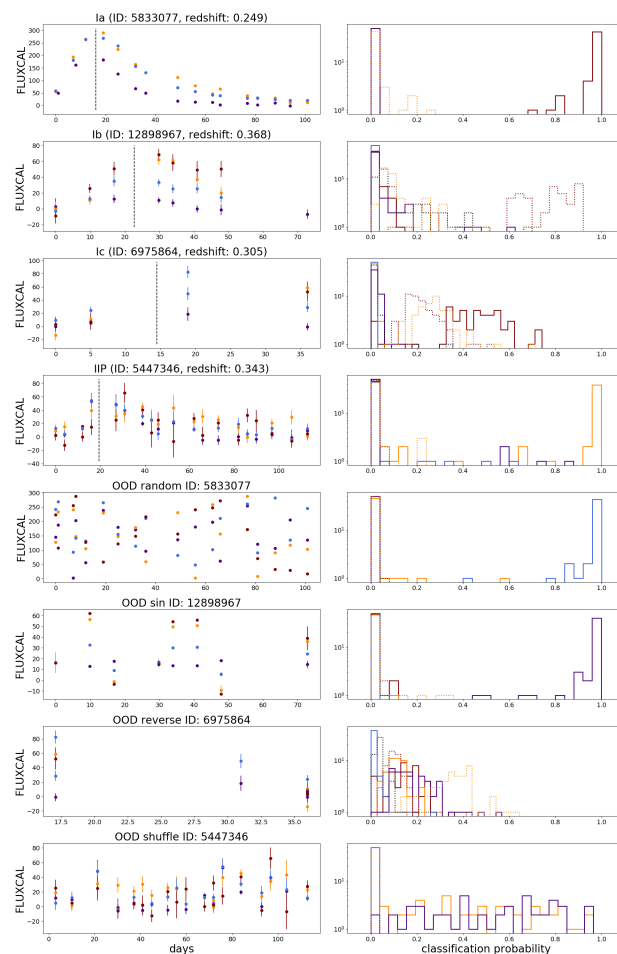

Figure 9. Out-of-distribution and SN light-curve prediction distribution for best variational configuration without redshift information for seven-way classification. First column depict light-curves, second column their classification probabilities. First four rows show SNe light-curves while last four rows are out-of-distribution events. For random and sinusoidal light-curves the classification probabilities are concentrated, giving a misleading high-certainty of classification and therefore entropy. The expected behaviour for out-of-distribution predictions is better represented by shuffle and reverse events shown in the last row. These events have either low predicted probabilities or large uncertainties in the predictions.

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 SNIa, m_B is the rest-frame B magnitude (at peak luminosity), x_1 is the stretch parameter and c is the color. 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 supernovae 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 out-of-distribution 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 Figure 10 for both correctly classified type Ia supernovae and incorrectly classified core-collapse supernovae. The contribution by the latter represents less than 5% of the sample and peaks at redshifts between $0.4 - 0.5$.

Such contamination can be reduced by: using additional information such as host-galaxy redshifts, raising the probability threshold for sample selection as shown in Figure 10 or eliminating classified events with large uncertainties. We have shown that adding redshifts allows us to obtain a sample contaminated by $< 2\%$ core-collapse SNe, contamination which 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 Variational implementation, setting the threshold to > 0.9 increases the sample purity by reducing the number of miss-classified core-collapse SNe by a factor of four and only reducing the correctly classified SN Ia by $< 5\%$.

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 number 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 Figures 10 and 11. In spectroscopically selected samples, there is a large difference between its properties and the ones of the expected supernovae population. This is mainly due to spectroscopic selection effects (Kessler et al. 2018; D'Andrea et al. 2018). In our photometrically classified sample using no redshift information, we see almost constant selection efficiency throughout redshift. Importantly, color and stretch distributions match very closely the parent population. Asides from this selection bias reduction, we are able to classify type Ia supernovae at redshifts > 0.9 which is extremely difficult when using spectroscopy.
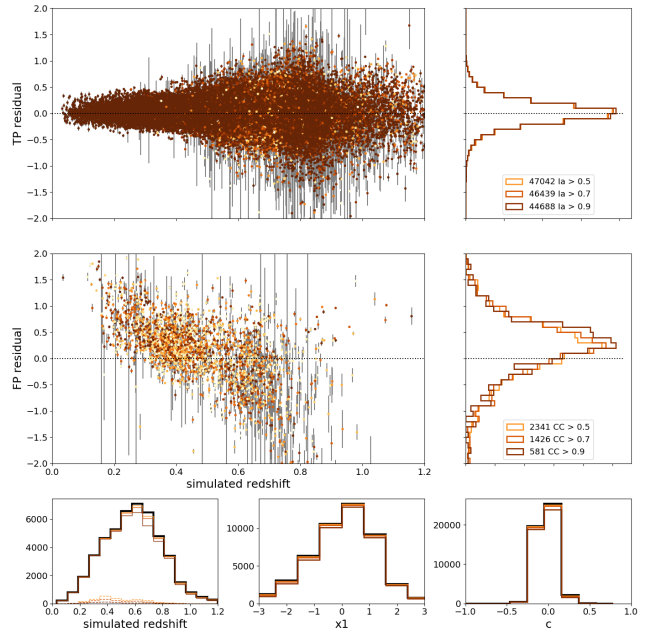

Figure 10. Hubble Residuals as a function of simulated redshift for the photometrically classified SN Ia sample using our best performing BBB RNN trained with the *complete* dataset with no redshift information. Top left: Hubble Residuals for correctly classified type Ia supernovae (TP) as a function of simulate redshift. Colors 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). Center left and right: same plots as above but for incorrectly classified core-collapse which represent the sample contamination. Bottom from left to right: redshift, color (c) and stretch (x1) distributions of the complete simulated Ia sample (black), selected TP at different probability thresholds (colored 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, color and stretch distributions and the contamination contribution $< 5\%$. Clearly, such photometric sample significantly reduces selection biases usually seen in spectroscopically selected samples.

6 SUPERNNOVA 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 8-core Intel(R) Core(TM) i7-7700K CPU (4.20GHz) with 15GB of RAM and an NVIDIA GeForce 1080Ti GPU to speed up training and inference.

Database construction, which includes formatting and normalization, is carried out once for the whole dataset 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 rate, without cyclic training times increase. For the Baseline RNN we find that training a single epoch with approximately 5,000 light-
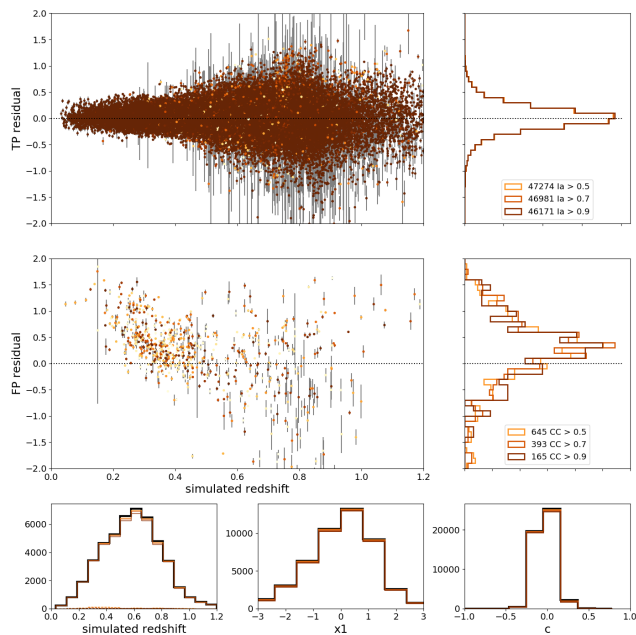
Figure 11. Hubble Residuals as a function of simulated redshift for the photometrically classified SN Ia sample using our best performing BBB RNN with photometric host-galaxy redshift information. Figure is similar to 10. Note that the level of contamination obtained in this sample is less than 2% which is equivalent to that expected of spectroscopically classified samples. As in the previous Figure, our selected sample probes almost completely the whole redshift, color and stretch parent population.

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	Number of SNe per second	device
Baseline	23,280	GPU
Baseline	2,820	CPU
Variational	342	GPU
Variational	58	CPU
Bayesian	264	GPU
Bayesian	56	CPU

curves ranges from 1 minute (GPU) to 10 minutes (CPU). Training the Baseline RNN to convergence requires from one and a half hours (GPU) to 15 hours (CPU). For the same light-curve numbers per epoch, Variational RNNs require slightly longer to train with 1.5 minutes (GPU), while BBB RNNs require 2 minutes 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 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.

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 vs. non type Ia supernovae with accuracies up to 95% for whole light-curves. When host-galaxy information is provided, accuracies increase to 98%. SuperNNova can be used in various classification tasks such as ternary and seven-way classification. For these classification tasks, accuracies are found to be above 90% when host-galaxy information is provided. By selecting photometric samples, SuperNNova allows the exploration and characterization of diverse astrophysical objects.

With the advent of large surveys discovering thousands of transients every night, it will be imperative to prioritize follow-up using partial light-curve classification. SuperNNova achieves 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% without redshift information and higher than 90% when host-galaxy redshifts are provided. Classification becomes harder when classifying into many targets (3 to 7), with accuracies only reaching around 63% near maximum light. For this classification task, the addition of host-galaxy redshifts can yield to 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 analyzed 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 dataset and emphasize the need for calibration diagnosis. We find that our RNNs are better calibrated than a Random Forest algorithm applied to the same dataset.

In SuperNNova, we adopt a Bayesian view of neural networks, showing much improved uncertainty estimates, calibration and out of sample behavior 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 $< 2\%$, 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, 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 out-of-distribution, 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 supernovae classes. We expect to continue applying SuperNNova and improving it to a wide variety of classification problems.

ACKNOWLEDGEMENTS

REFERENCES

Astier P., et al., 2006, Astron.Astrophys., 447, 31
Astropy Collaboration et al., 2013, A&A, 558, A33
Bahdanau D., Cho K., Bengio Y., 2014, preprint, (arXiv:1409.0473)
Bazin G., Ruhlmann-Kleider V., Palanque-Delabrouille N., Rich J., Aubourg E., et al., 2011, Astron.Astrophys., 534, A43
Bellm E. C., 2018, preprint, p. arXiv:1802.10218 (arXiv:1802.10218)
Bernstein J. P., et al., 2012, Astrophys. J., 753, 152
Betoule M., et al., 2014, Astron.Astrophys., 568, A22
Blundell C., Cornebise J., Kavukcuoglu K., Wierstra D., 2015, preprint, (arXiv:1505.05424)
Campbell H., et al., 2013, ApJ, 763, 88
Charnock T., Moss A., 2017, ApJ, 837, L28
Chung J., Gulcehre C., Cho K., Bengio Y., 2014, preprint, p. arXiv:1412.3555 (arXiv:1412.3555)
D'Andrea C. B., et al., 2018, preprint, p. arXiv:1811.09565 (arXiv:1811.09565)
Dai M., Kuhlmann S., Wang Y., Kovacs E., 2018, MNRAS, 477, 4142
DeGroot M. H., Fienberg S. E., 1983, Journal of the Royal Statistical Society. Series D (The Statistician), 32, 12
Foley R. J., et al., 2018, MNRAS, 475, 193
Fortunato M., Blundell C., Vinyals O., 2017, preprint, p. arXiv:1704.02798 (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, p. arXiv:1506.02142 (arXiv:1506.02142)
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, preprint, (arXiv:1811.02381)
Hlozek R., et al., 2012, ApJ, 752, 79
Hochreiter S., Schmidhuber J., 1997, Neural Comput., 9, 1735
Hunter J. D., 2007, Computing In Science & Engineering, 9, 90
Ishida, E. E. O. and de Souza, R. S. 2013, MNRAS, 430, 509
Ishida E. E. O., et al., 2018, MNRAS,
Jones D. O., et al., 2018, ApJ, 857, 51
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, Publ. Astron. Soc. Pac., 122, 1415
Kessler R., et al., 2018, preprint, (arXiv:1811.02379)
Kimura A., Takahashi I., Tanaka M., Yasuda N., Ueda N., Yoshida N., 2017, preprint, p. arXiv:1711.11526 (arXiv:1711.11526)
Kingma D. P., Ba J., 2014, preprint, (arXiv:1412.6980)
Kuznetsova N. V., Connolly B. M., 2007, ApJ, 659, 530
LSST Science Collaboration Abell P. A., Allison J., Anderson 2009, preprint, p. arXiv:0912.0201 (arXiv:0912.0201)
LeCun Y., Cortes C., 2010, http://yann.lecun.com/exdb/mnist/
Lochner M., McEwen J. D., Peiris H. V., Lahav O., Winter M. K., 2016, ApJS, 225, 31
McKinney W., 2010, in van der Walt S., Millman J., eds, Proceedings of the 9th Python in Science Conference. pp 51 – 56
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, Journal of Cosmology and Astro-Particle Physics, 2016, 008
Moss A., 2018, preprint, (arXiv:1810.06441)
Narayan G., et al., 2018, The Astrophysical Journal Supplement Series, 236, 9
Niculescu-Mizil A., Caruana R., 2005, in Proceedings of the 22Nd International Conference on Machine Learning. ICML '05. ACM, New York, NY, USA, pp 625–632, doi:10.1145/1102351.1102430, http://doi.acm.org/10.1145/1102351.1102430
Paleari F., Galvani T., Amara H., Ducastelle F., Molina-Sánchez A., Wirtz L., 2018, 2D Materials, 5, 045017
Paszke A., et al., 2017, https://github.com/pytorch/pytorch
Pedregosa F., et al., 2011, Journal of machine learning research, 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
The Astropy Collaboration et al., 2018, preprint, (arXiv:1801.02634)

The PLAsTiCC team et al., 2018, preprint, (arXiv:1810.00001)

Van Der Walt S., Colbert S. C., Varoquaux G., 2011, preprint, p. arXiv:1102.1523 (arXiv:1102.1523)

Waskom M., et al., 2018, Seaborn v0.9.0, doi:10.5281/zenodo.1313201, https://doi.org/10.5281/zenodo.1313201

APPENDIX: REPRODUCIBILITY

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

APPENDIX: BEHAVIOUR OF OOD IN MNIST

To better understand the behaviour of Bayesian Neural Networks, we carried out two experiments on the MNIST dataset (LeCun & Cortes 2010). This dataset contains images of handwritten digits.

First, we trained three (respectively standard, variational and bayesian) feedforward networks to classify flattened digit images into one of ten classes, each representing a digit between 0 and 9. To evaluate performance on out-of-distribution data, we asked the network to carry out predictions on rotated digits, as well as completely randomized images. In Figure 1, we show the predictions for the standard feedforward 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 Variational and Bayesian networks, we observe a much more interesting behaviour (Figures 2 and 3). When the rotation is small, the network behaves in much the same way as a non bayesian one and gives max 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 the image bears some similarity to the data which 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 out-of-distribution samples which yields a large entropy.

Second, we trained three recurrent neural networks (respectively standard, variational and bayesian), 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 feedforward counterpart (Figure 4) 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 exhibited by the bayesian feedforward networks, and that extensive parameter tuning was required. The Variational and Bayesian networks (Figures 5 and 6) show, as before, an increased variance in the prediction probabilities for random images. However, the behaviour on random data differs significantly. The variational
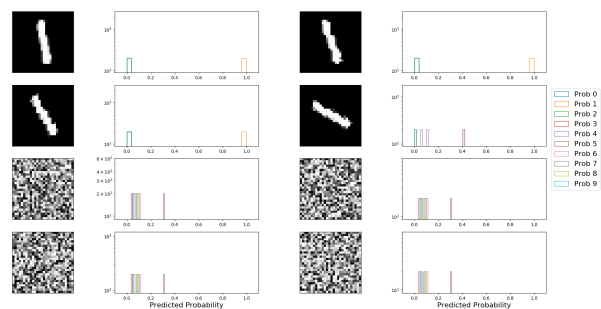
Figure 1. A standard feedforward network on MNIST. On the first and third column, images show the input given to the network. The classification probabilities for each target are given as histograms in the second and third columns.
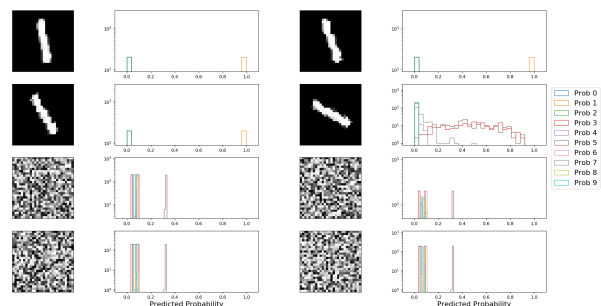

Figure 2. A Variational Dropout feedforward network on MNIST. Note the similar behaviour for OOD events when compared to Figure 1.
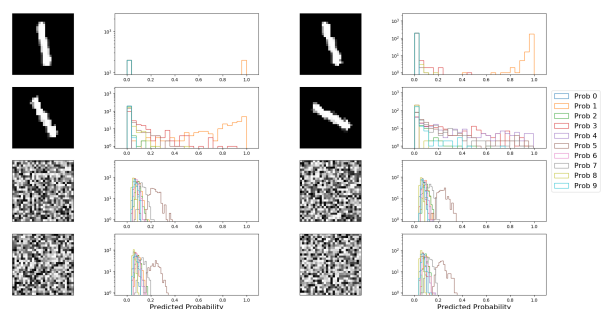

Figure 3. A Bayesian By Backprop feedforward network on MNIST. Note the similar behaviour for OOD events when compared to Figure 1.

network seems to have collapsed on predicting a single class with high probability while the bayesian network exhibits large variance for multiple classes.

While we have verified that tuning the various hyperparameters improves the uncertainty performance on this qualitative examination, it is clear that the behaviour of Bayesian recurrent networks should be critically analyzed: 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
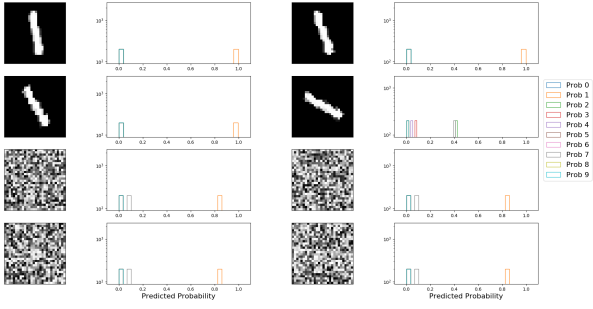
Figure 4. A recurrent network on MNIST. This RNN is able to obtain similar prediction behaviour as Figure 1 which is what is expected for OOD events.
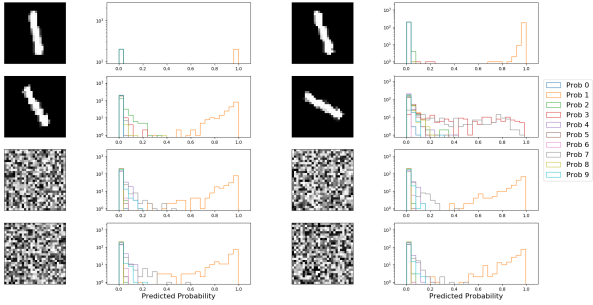

Figure 5. A Variational Dropout recurrent network on MNIST. This network collapses and outputs high-certainty predictions for OOD images.
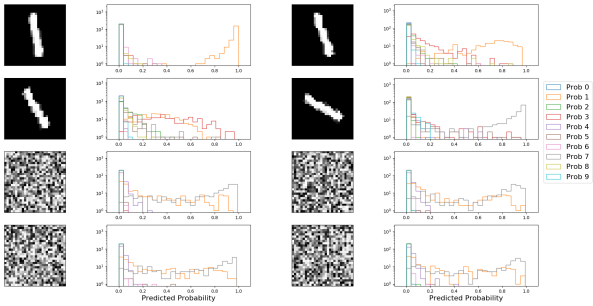

Figure 6. A Bayesian By Backprop recurrent network on MNIST. This network collapses and outputs high-certainty predictions for OOD images.

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 artifacts even after normalization. Future work will focus on characterizing this phenomenon and developing methods to improve robustness on out-of-distribution data.

This paper has been typeset from a TeX/LaTeX file prepared by the author.