The Astrophysical Journal Letters, 937:L15 (10pp), 2022 September 20 https://doi.org/10.3847/2041-8213/ac8f4b

Applications of Machine Learning to Predicting Core-collapse Supernova Explosion Outcomes

Benny T.-H. Tsang[1,2], David Vartanyan[1], and Adam Burrows[3]
[1] Department of Astronomy and Theoretical Astrophysics Center, University of California, Berkeley, CA 94720, USA; benny.tsang@berkeley.edu
[2] Kavli Institute for Theoretical Physics, University of California, Santa Barbara, CA 93106, USA
[3] Department of Astrophysical Sciences, Princeton University, NJ 08544, USA
Received 2022 August 1; revised 2022 August 26; accepted 2022 September 4; published 2022 September 22

Abstract

Most existing criteria derived from progenitor properties of core-collapse supernovae are not very accurate in predicting explosion outcomes. We present a novel look at identifying the explosion outcome of core-collapse supernovae using a machine-learning approach. Informed by a sample of 100 2D axisymmetric supernova simulations evolved with FORNAX, we train and evaluate a random forest classifier as an explosion predictor. Furthermore, we examine physics-based feature sets including the compactness parameter, the Ertl condition, and a newly developed set that characterizes the silicon/oxygen interface. With over 1500 supernovae progenitors from $9-27\ M_{\odot}$, we additionally train an autoencoder to extract physics-agnostic features directly from the progenitor density profiles. We find that the density profiles alone contain meaningful information regarding their explodability. Both the silicon/oxygen and autoencoder features predict the explosion outcome with ≈90% accuracy. In anticipation of much larger multidimensional simulation sets, we identify future directions in which machine-learning applications will be useful beyond the explosion outcome prediction.

Unified Astronomy Thesaurus concepts: Core-collapse supernovae (304); Astronomy data analysis (1858); Astrostatistics techniques (1886); Classification (1907); Random Forests (1935); Convolutional neural networks (1938)

1. Introduction

Machine learning (ML) has become an integral part of astrophysics research in the recent decade (Ball & Brunner 2010; Ivezić et al 2014; Fluke & Jacobs 2020). In essence, ML systems are computational tools that are efficient in assimilating complex probability distributions. These distributions are ubiquitous in both observational and theoretical astronomy. For example, characteristic separation of data samples in the image domain has facilitated reliable classification of galaxies (Aniyan & Thorat 2017; Cheng et al. 2020). Similar success has been achieved in the time domain for variable star classification (Naul et al. 2018; van Roestel et al. 2021). In addition, identification of outliers from data clusters of known types, a technique known as novelty or anomaly detection, enables the discovery of previously unknown objects and new classes of objects (Tsang & Schultz 2019; Williamson et al. 2019; Villar et al. 2020; Ishida et al. 2021; Malanchev et al. 2021; Bengyat & Gal-Yam 2022).

Beyond classification and detection, one can regard multi-physics simulation products themselves as the complex distributions to be learned. Emulations of computationally costly simulations can be generated quickly by sampling new data points in the latent spaces that are trained to embody the fully-fledged simulations (Caldeira et al. 2019; Mustafa et al. 2019; Vogl et al. 2020; Horowitz et al. 2021). Moreover, ML systems are powerful tools for parameter inference, connecting observables to physical parameters that are oftentimes degenerate (Ksoll et al. 2020; Villanueva-Domingo et al. 2022; Villanueva-Domingo & Villaescusa-Navarro 2022). Parameters can even be distributions themselves, e.g., the equation of state of neutron stars (Krastev 2022), the tensor closure for neutrino transport (Harada et al. 2022), and turbulence closures for subgrid modeling (Karpov et al. 2022).

However, entirely lacking is the application of ML techniques to predicting core-collapse supernovae (CCSNe) explosion outcomes. CCSNe simulations are computationally expensive endeavors in both human and machine terms, and thus are a ripe opportunity for ML application. The explosion mechanism of CCSNe through the neutrino-heating mechanism has been studied as a computational problem for more than half a century (Colgate & White 1966; Bethe & Wilson 1985), through both detailed computational simulations and much cruder prescriptive methods (e.g., imposing a thermal bomb, driving a piston, or other rudimentary prescriptions). Only in the last decade have multidimensional simulations become the mainstay, with various groups performing scores of two-dimensional axisymmetric computations and tens of 3D simulations. Though population suites of CCSNe have been evolved in 2D (Ertl et al. 2016; Summa et al. 2016; Radice et al. 2017; Burrows et al. 2018; Vartanyan et al. 2018; Burrows & Vartanyan 2021; Kuroda et al. 2022) and, more selectively, 3D (O'Connor & Couch 2018a; Summa et al. 2018; Burrows et al. 2019; Glas et al. 2019; Nagakura et al. 2019; Vartanyan et al. 2019; Kuroda et al. 2020; Burrows et al. 2020; Obergaulinger & Aloy 2021; Vartanyan et al. 2022), developing hundreds, let alone thousands, of 3D simulations may not be feasible in the coming decade.

To circumvent this limitation, and in order to explore the explosion landscape by progenitor for final explosion energies, observational signatures, and nucleosynthetic compositions, various groups have developed CCSNe population studies using simplified prescriptions in reduced dimensions. Different such approaches include analytical approximations of

protoneutron star cooling (Ugliano et al. 2012), PUSH (Perego et al. 2015; Curtis et al. 2021), simple pistons (Sukhbold et al. 2016), and spherically symmetric turbulence models (STIR; Mabanta et al. 2019; Couch et al. 2020), often calibrated to SN1987a and the Crab and comparing the derived explosion outcomes with various formulated predictions (e.g., the antesonic condition, Pejcha & Thompson 2012; Raives et al. 2018; the Ertl criterion, Ertl et al. 2016; a semianalytical pre-SN parameterization, Müller et al. 2016a).

These methods rely on simplifying approximations for both explosion modeling and explosion prediction. In light of this, the motivation of our paper is to present a summary overview of potential ML approaches to CCSNe outcome prediction as a proof of concept of the eventual goal—developing ML techniques, trained on the results of extant multidimensional simulations, to predict explosion outcomes while circumventing costly detailed simulations. Our intent here is not to be comprehensive, but rather to present a sample of the applicable methods and to galvanize the use of these techniques more broadly in the community. We wish to highlight the versatility and potential future use of ML, and identify potential difficulties and obstacles.

In Section 2, we describe our methodology including the simulated data set (Section 2.1), the various physics-based explosion conditions (Section 2.2), an unsupervised feature extraction approach used to derive physics-agnostic explosion criteria (Section 2.3), a baseline random forest (RF) classifier used as an explosion outcome predictor (Section 2.4), and a semisupervised label propagation approach (Section 2.5). In Section 3, we present the results comparing the accuracy of the various features in predicting explosion outcomes. We summarize our conclusions in Section 4 and identify future directions in Section 5.

2. Methods

Our goal is to survey various machine-learning approaches in tandem with a selection of explosion criteria to study their value in predicting explosion outcomes ab initio. These explosion outcome predictors are trained and tested on a suite of 100 2D axisymmetric CCSNe simulations run with the radiation-hydrodynamic code Fornax. Fornax (Skinner et al. 2019) is a multidimensional, multigroup code constructed to study CCSNe. It features an M1 solver (Vaytet et al. 2011) for neutrino transport with detailed neutrino microphysics and an approximation to general relativity (Marek et al. 2006).

2.1. Data Sets

We selected a subset of 100 initial progenitor models from 9 $-26.99\ M_\odot$ to evolve in 2D-axisymmetry (D. Vartanyan et al. 2022, in preparation) using Fornax for typically 1 s after core bounce to ascertain their explodability (discussed in more detail in Wang et al. 2022). These models were evolved with neutrino heating as the explosion mechanism, absent rotation, and magnetic fields. The models had a resolution of 1024×256 in r, θ with outer radii extending from 30,000 km for the lower-mass stellar progenitors and to 100,000 km for the most-massive progenitors. These models were chosen to be representative as much as possible of the Salpeter initial mass function. They were selected to span broadly the distribution in density profiles, compositional interfaces, compactness, and μ_4/M_4 (discussed below). We categorize explosion as a runaway shock radius within the simulation time. Of the 100 models, 64 exploded, and 36 did not. A runaway shock radius is not a guarantee of explosion, as that depends on unbinding the stellar envelope, which can having binding energies of a few tenths of a Bethe exterior to the simulation grid. Resolving this requires late-time simulations. Indeed, in Burrows & Vartanyan (2021), a subset of the progenitors studied were evolved out to approximately 4 s after bounce, and models that had runaway shock radii there all resulted in net explosions that unbound the entire envelope and yielded up to several Bethe of explosion energies. These 100 models with known explosion outcomes based on the 2D simulations will be referred to as the *labeled* data set. Our 100 models were selected from the newest stellar progenitor models in Sukhbold et al. (2016, 2018). The compilation contains 12 progenitors in the mass range of $9-11.75\ M_\odot$ in increments of $0.25\ M_\odot$ (from Sukhbold et al. 2016), and 1500 progenitors in the range of $12-26.99\ M_\odot$ in increments of $0.01\ M_\odot$, for a grand total of 1512 stellar progenitors. The 1412 progenitor models that were not evolved in Fornax, and therefore do not have known explosion outcomes, are referred to as the *unlabeled* data set. All the models studied were evolved as single-star progenitors, absent binary effects. We note that we are limited by the data set size of this ML exercise, as well as by the complexity of the physical phase space explored.

2.2. Physics-based Features

Due to the limited number and high dimensionality of the progenitor models, explosion outcome predictors in the form of binary classifiers cannot be well trained using the raw stellar profiles as inputs. Instead, parameters of much lower dimension, known as *features*, are obtained to represent the distinctive characteristics of the models in a process known as *feature extraction*. Multiple attempts have been made to identify such explosion conditions, often ab initio, that can serve to predict CCSNe outcomes (e.g., O'Connor & Ott 2011; Pejcha & Thompson 2012; Dolence et al. 2015; Müller et al. 2016a; Ertl et al. 2016; Summa et al. 2018; Gogilashvili & Murphy 2022). These derive in heritage from some variation on the concept of a critical condition (Burrows & Goshy 1993), which suggests a relation between neutrino luminosity and mass accretion at the shock, above which unabated shock expansion concludes in explosion. Below, we summarize three types of explosion metrics whose utility in predicting explosion outcomes we explore with our ML approaches. We focus on compactness and the Ertl condition because of their widespread use, their relative simplicity, and their ab initio nature. We also target an additional feature—the role of the silicon–oxygen compositional interface.

2.2.1. Compactness

The compactness parameter characterizes the core structure and is defined as (O'Connor & Ott 2011):

$$\xi_M = \frac{M/M_\odot}{R(M)/1000\ \mathrm{km}}, \tag{1}$$

where the subscript M denotes the interior mass coordinate at which the compactness parameter is evaluated. For our purposes, we evaluate the compactness parameter $\xi_{1.75}$ at $M = 1.75\ M_\odot$, generally encompassing the Si/O interface for many the progenitor models. The compactness is often used as

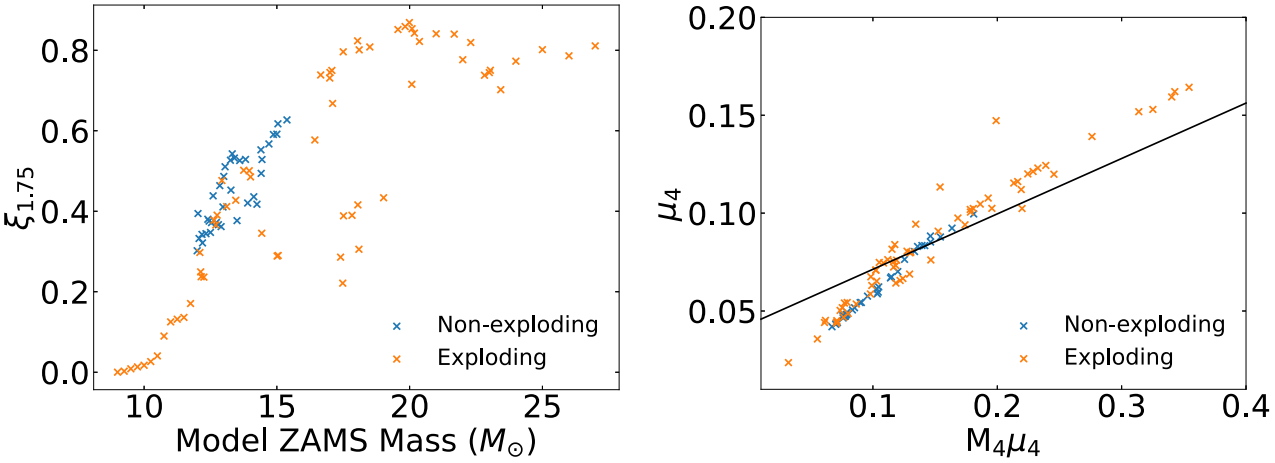

Figure 1. The compactness $\xi_{1.75}$ vs. ZAMS mass (left) and the Ertl parameterization (right) for the 100 labeled models evolved in 2D. Blue crosses indicate nonexploding models, and orange crosses exploding models. The putative separation curve in the Ertl parameterization does not separate explosion outcomes reliably, with exploding models both above and below the line, although nonexploding models are almost entirely below the line (Wang et al. 2022). Note the clustering of nonexploding models in both figures, particularly between 12 and 15 $M_\odot$ with different outcomes vis-á-vis explosion for a given physical parameter (see also Sukhbold et al. 2018), perhaps indicative of a third dimension necessary to break the degeneracy.

an ab initio explosion condition because it depends only on the progenitor properties. While higher compactness is correlated with higher luminosities, accretion rates, and remnant masses (O'Connor & Ott 2013), compactness does not readily lend itself as an explosion condition, and suggestions that explosion is inhibited above a certain compactness parameter are false (Burrows et al. 2020; with the exception that massive models may initially drive a successful shock, but then later implode into a black hole due to the large gravitational binding energy). We plot in the left panel of Figure 1 the distribution of compactness versus the zero-age main-sequence (ZAMS) mass of the labeled data set, with exploding models indicated in orange and nonexploding in blue. For most progenitors the mass at which ξ_M is calculated usually encompasses the Si/O interface entropy and density jump, and this is discussed below.

2.2.2. Ertl Parameter

The Ertl condition for explosion (Ertl et al. 2016) is another ab initio explosion condition. It identifies a μ_4 and $\mu_4 \times M_4$ space, where μ_4 is a measure of the slope of the mass density at an entropy of 4 (per baryon per Boltzmann's constant) and M_4 is the interior mass at that entropy. This approximately corresponds to the location of an entropy/density jump at the Si/O interface. The Ertl condition purports to be a statement of criticality, with μ_4 and $\mu_4 \times M_4$ relating indirectly to L_ν and $\dot{M}$, the neutrino luminosity and the mass accretion rate. We show in the right panel of Figure 1 the Ertl curve suggested to separate explosion and nonexplosion, overplotted with the results of our 100 2D simulations (see also Wang et al. 2022). We note the poor agreement between our simulation results and the Ertl prediction, and comment on this more in Section 3.

2.2.3. Si/O Interface Parameters

Lastly, we posit a physically motivated explosion condition that looks at prominent density interfaces (often the Si/O interface; Wang et al. 2022) whose accretion by the shock surface can revive a stalled shock into successful explosion (Fryer 1999; Sukhbold et al. 2016; Burrows et al. 2018; Ott et al. 2018; Vartanyan et al. 2018; Burrows et al. 2019; Vartanyan et al. 2021; Burrows & Vartanyan 2021; Boccioli et al. 2022). A sharp drop in density translates into an immediate drop in ram pressure at the shock surface upon encountering this interface, whereas the accretion-powered luminosity interior to the shock is sustained for an advective timescale (Wang et al. 2022). This drop in ram pressure, while maintaining a higher luminosity, promotes explosion and may be key to explosion for massive stars. Lower-mass models of $\approx 9-10$ $M_\odot$ may explode simply on the virtue of their very steep density profiles. We identify the location in mass coordinate $M_{\rm SiO}$ and the magnitude of the density jump across such interfaces $\Delta\rho_{\rm SiO}$ in all 1512 models in the full progenitor data set. For each of the models studied here, we identified the Si/O or other prominent interface by looking for the steepest drop in density in the stellar progenitor profile exterior to the iron core. The stellar density may drop by as much as $2-3$ times over less than 0.01 $M_\odot$.

The extraction of the interface features can be complicated by the presence of multiple, fragmented burning shells (see also Vartanyan et al. 2021; Laplace et al. 2021 for a similar conclusion regarding the density profiles of binary stars). Sukhbold et al. (2018) identify multiple burning shells during late-stage stellar evolution, where the physics is poorly resolved and the results prone to stochasticity (see also Wang et al. 2022). Merging of the two shells into a single steeper shell will produce a more prominent density drop conducive to successful core-collapse explosions. We plot in Figure 2 the Si/O interface mass coordinate versus the magnitude of the density drop from the labeled data set. We see a nonuniform distribution of both compactness (in Figure 1) and the Si/O interface with clustering and multiple branches (see also Müller et al. 2016a; Sukhbold et al. 2018; Wang et al. 2022), as well as multivalued outcomes (explosion and not) in a small range of the plotted phase space. Using the Si/O interface yields a clearer delineation between explosion and nonexplosion than compactness, but in both cases we see degeneracy in the outcome for the given putative explosion criteria.

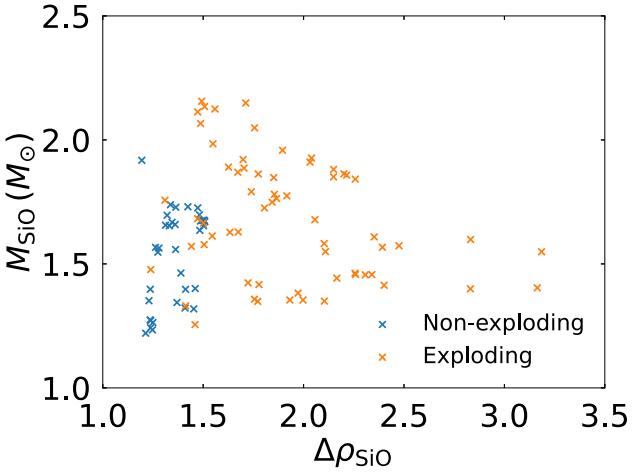

Figure 2. The Si/O interface parameter distribution of the 100 labeled models evolved in 2D. Blue crosses indicate exploding models, and orange crosses nonexploding models. Note the clumping in phase space for weaker interfaces located deeper in.

2.3. Unsupervised Feature Extraction

The physics-based feature sets described in Section 2.2 are not always very effective. Modern autoencoder neural network architectures offer an alternate data-driven, physics-agnostic approach to extracting relevant features in an unsupervised manner. Autoencoders consist of two main components: an encoder and a decoder. The encoder is designed to take an input vector and convert it into a feature vector of much lower dimension. The decoder, on the other hand, attempts to reconstruct the input from the feature vector. By training the encoder–decoder pair to match the input and the reconstruction, the autoencoder learns to capture the important information in the input without human intervention. Example usage in astronomy includes variable star (Naul et al. 2018; Tsang & Schultz 2019) and galaxy (Portillo et al. 2020) classification, anomaly detection for supernova light curves (Villar et al. 2020), detection of strong lensing features in images (Cheng et al. 2020), and the denoising of radio images (Gheller & Vazza 2022). Autoencoders essentially serve as an apparatus for data compression, allowing ML systems to operate more effectively on the much-lighter-weight feature vectors rather than the raw inputs.

Here, we explore the application of autoencoders to extracting representative features directly from the density profiles of the stellar progenitor models. To this end, we implement a basic autoencoder network in PYTORCH (Paszke et al. 2019). We adopt three 1D convolutional layers (CONV1D) as the main components of the encoder. The convolutional layers are designed to preserve the spatial information of the mass distribution in the stellar density profiles. The decoder is constructed using three corresponding CONVTRANSPOSE1D layers. The hyperbolic tangent function (tanh) is used as the nonlinear activation function after each CONV1D and CONVTRANSPOSE1D layer. After passing through the encoder's final layer, the resultant vector is commonly known as the *embedding*, which is of much smaller dimension than the input sequence. The embedding vectors can be regarded as the reduced-dimension feature vectors that can be used for other downstream tasks. By construction, the tanh activation function produces embedding vectors $z \in [-1, 1]^{d_z}$, where d_z is the embedding dimension. In our case study, we focus on the explosion outcome prediction task, which is set up as binary classification. The autoencoder architecture is presented in the schematic diagram in Figure 3.

To explore the learning capacity of the autoencoder, we vary the dimension of the embedding vector by adjusting the strides of the convolutional layers, covering embedding dimensions of $d_z = 2$ to 32 in factors of 2. The number of total trainable parameters (weights and biases) depends solely on the kernel and filter sizes. Since we do not vary those network parameters, the autoencoders we use in this work contain a constant number of trainable parameters of 106. The kernel size, stride, and the breakdown of the number of parameters in each layer can be found annotated in Figure 3. We keep the network size to a minimum to highlight the utility of a simple autoencoder.

We use the 1412 unlabeled models as the training set of the autoencoder models. In other words, the autoencoder is only tasked to learn the representation of the density profiles without regard to their explodability. We truncate the density profiles and consider only mass coordinates between $M_{\rm min} = 1\,M_\odot$ and $M_{\rm max} = 2.3\,M_\odot$. Interior to $M_{\rm min}$, matter collapses onto the protoneutron star and lies interior to the stalled shock surface. On the high mass end, it is rare for relevant interfaces in the studied ZAMS distribution to exist beyond $M_{\rm max}$ and still accrete on relevant timescales for neutrino-driven CCSNe. The density profiles of the KEPLER models with different ZAMS masses (Sukhbold et al. 2016, 2018) used as FORNAX supernova progenitors vary in grid resolution between $M_{\rm min}$ and $M_{\rm max}$, ranging typically from 800 to 1200 zones. To standardize the dimension of the input profiles, we interpolate and rebin the logarithm of the truncated density profiles onto a uniform linear mass grid with $N_m = 128$ points. The reduced dimension of 128 is adequate in capturing the sharp jumps in density in the progenitor density profiles (see the solid lines in the top panels of Figure 4). To isolate trends, we subtract the means from the profile segments and normalize them independently. Mathematically, for each progenitor, the normalized density profiles take the form

$$\hat{x}_m = (x_m - \langle x_m \rangle)/(\max(x_m) - \min(x_m)), \tag{2}$$

where m is the integer index of the uniform mass grid, $x_m = \log_{10}(\rho_m)$ is the logarithmic mass density of the mth mass bin, and $\langle \cdot \rangle$ denotes the mean value over the mass grid. The 128-dimension density profile segments $\{\hat{x}_m\}$ are used as inputs to the autoencoders. By rescaling and normalizing the density profiles, we set up the autoencoder to focus on the *shape* and omit the *absolute scale* of the density profiles. Similar to Wang et al. (2022), we gathered that the change in density/ram pressure is more relevant for the explosion outcome than the absolute density value. Mean squared error (MSE) between the input and the reconstructed density profiles $\{\tilde{x}_m\}$ is used as the loss function for training: $L_{\rm AE} = \sum_m (\hat{x}_m - \tilde{x}_m)^2/N_m$.

Weights of the convolutional layers are initialized using the `kaiming_normal_` initializer (He et al. 2015) in PYTORCH. Biases are initialized as zeros. Optimization is done using the ADAM optimizer (Kingma & Ba 2014). Training is conducted with a constant batch size of 100 and a learning rate of 10^{-2} for 500 epochs. Since our key goal is to present the utility of physics-agnostic features, we did not perform a systematic hyperparameter study to optimize the autoencoder. Due to the limited size of the labeled data set, we also did not attempt to

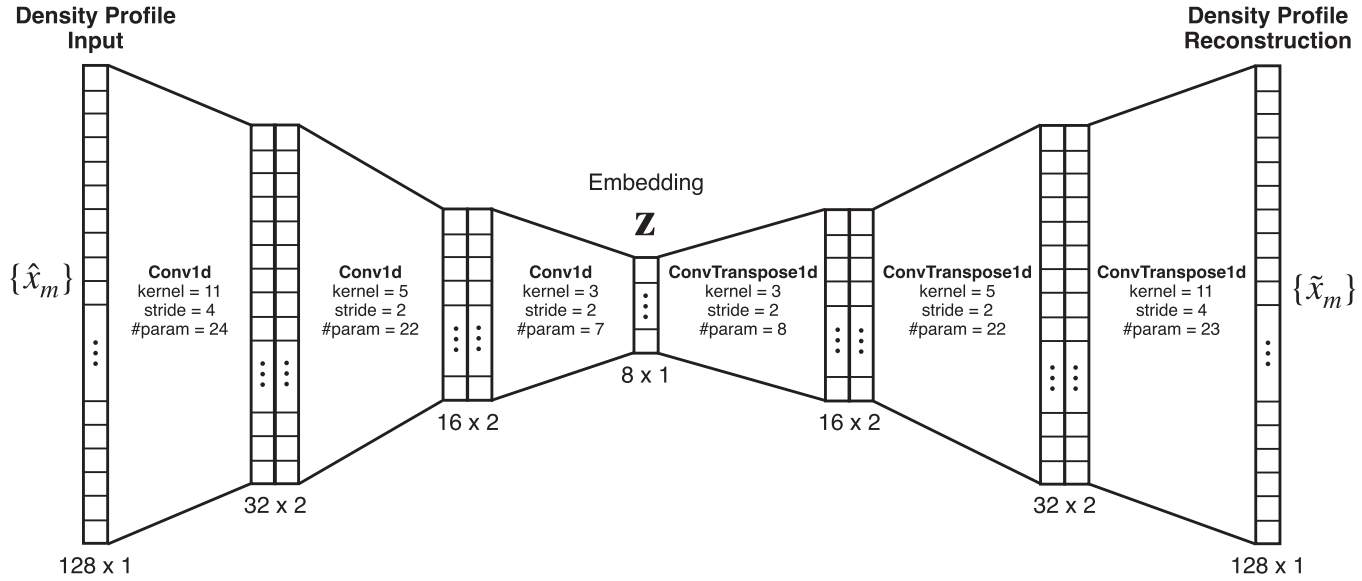

Figure 3. Schematic diagram of the autoencoder's neural network architecture. It employs three basic CONV1D layers as the encoder, converting the input density profile sequence $\{\hat{x}_m\}$ into the reduced-dimension embedding vector z. The decoder similarly utilizes three CONVTRANSPOSE1D layers, trained to produce a reconstruction of the input density profile $\{\tilde{x}_m\}$. The hyperparameters, number of trainable parameters, and the dimension for each layer are annotated.

train the autoencoder simultaneously with a binary classifier for explosion outcome prediction. These will be instructive follow-up studies when more comprehensive data sets are available.

2.4. Classifier Training

Using the features obtained in Sections 2.2 and 2.3, we train explosion outcome predictors in the form of binary classifiers. We adopt the SKLEARN implementation of RANDOMFOREST-CLASSIFIER as a common classifier baseline. During training, we adopt a five-fold, 80/20 split to divide the labeled data set (with 100 models) into training and testing sets. The training/testing partitions are generated using the STRATIFIEDKFOLD function of the SKLEARN package. A constant seed is used for the five-fold random split, resulting in the same partitions of training/testing data across classifiers trained with different feature sets. To allow fair comparisons between classifiers trained with different feature sets, we use a relatively simple RF setup with fixed parameters of `n_estimators = 5`, `criterion = 'gini'`, `max_depth = 3`, `min_smaple_leaf = 2`, and `max_features = 'sqrt'`. The RF classifiers therefore all have a fixed number of five decision trees. With the feature sets we explored in Section 3.2, our RF parameters lead to about 7−10 nodes per tree partitioning the feature spaces, or about 35–50 total nodes. We use the accuracy, precision, recall, and the F1 score to assess the performance of the classifiers.

2.5. Semisupervised Learning with Label Propagation

During training, classifiers often require sufficiently large data sets to sample the distributions of various object classes in the feature space. Labeled data sets, in our case multi-dimensional simulations, are costly to produce both in computer and human hours. Alternately, unlabeled data sets, in our application 1D progenitor models, are usually much cheaper to obtain. Semisupervised learning is a hybrid approach devised to use a limited sample of labeled data to assign mock labels to a larger, unlabeled data set based on some distance metrics in the feature space. The hope is that by propagating the labels to the larger unlabeled data set and incorporating it in training, the classifier can better learn the data distribution and achieve higher overall prediction accuracy.

Semisupervised approaches rely on a key presumption that the distributions of different classes are continuous, i.e., data samples close together in the feature space are likely to be of the same class. However, as we have seen in Figures 1 and 2, there are complex branches and overlaps associated with degeneracy and/or modeling stochasticity. Nevertheless, in this paper we attempt the label propagation technique to probe the potential utility of semisupervised learning approaches in improving explosion outcome prediction.

The procedure of our label propagation study is as follows. With each feature set selection and in each cross-validation split, we repeat the RF classifier training described in Section 2.4 after (i) randomly removing 50% or 75% of the known explosion outcomes in the training split (sized 80) and (ii) relabeling them based on the distances to their neighbors whose explosion outcomes are retained. In other words, we pretend that our labeled data set is 50%/75% smaller than it is and allow the label propagation algorithm to re-create a 80-model training set for the RF classifier. Evaluation of prediction performance is still done using the 20-model testing splits.

We employ the LABELSPREADING model in SKLEARN for this task. Fundamentally, LABELSPREADING works by building a fully connected graph connecting all the data samples and propagating labels based on the pairwise distances. To limit the scope of this exercise, we adopt a constant set of label propagation parameters. In particular, we use the K-nearest neighbor kernel as the distance metric (`kernel = ''knn''`) with `n_neighbors = 5`. A soft clamping factor of `alpha = 0.1` is used to allow the algorithm to change at most 10% of the retained labels from the samples to account for the stochasticity in the explosion simulations. Pseudo-labels are assigned to the samples with their explosion outcomes removed via the `transduction_` operation.

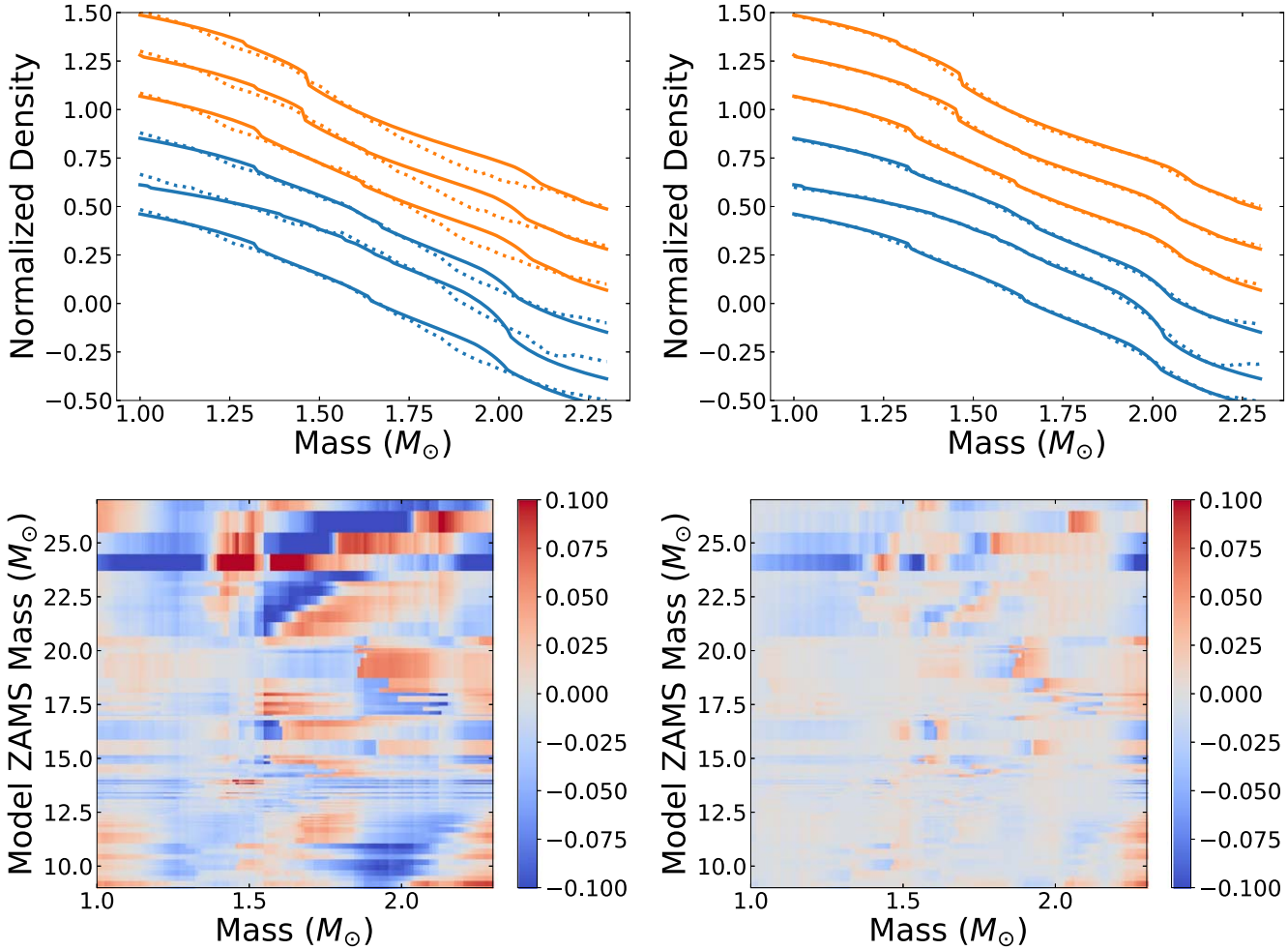

Figure 4. Top: examples of the input (solid) and the reconstructed (dotted) density profiles to and from the autoencoders with embedding dimensions of 2 (left) and 8 (right). Blue (orange) curves correspond to exploding (nonexploding) models. The profiles are plotted with consecutive vertical offsets of 0.1 for better visualization. Bottom: the reconstruction error (reconstruction − input) of the normalized density profile by the autoencoders with embedding dimensions of 2 (left) and 8 (right). The vertical axis denotes the ZAMS mass of all 100 models in the labeled data set. The fiducial model can encode density profiles with errors of $\lesssim$0.05.

3. Results

3.1. Autoencoder Performance

Even with the highly limited number of only 106 trainable parameters, the autoencoders with different embedding dimensions converge efficiently to an MSE loss of 10^{-4}–10^{-3} within less than 50 epochs. To visualize the representation performance of the autoencoders, we compare examples of the input and the reconstructed density profiles in the top panels of Figure 4. With $d_z = 2$, the reconstructed profiles miss some of the sharper interface transitions, but trace the overall trends of the profiles quite well. With $d_z \geqslant 8$, the density profiles are all well captured by the autoencoders. In the bottom panels of Figure 4, we show the (reconstruction − input) error from all the 100 models in the labeled data set. The maximum error is about 0.1 for the autoencoder with an embedding size of $d_z = 2$, whereas for $d_z = 8$ the typical deviations are less than about 0.05. We emphasize that the representation performance of the autoencoder architecture can likely be improved or fine-tuned with a more thorough study. To establish the efficacy of the physics-agnostic features, we choose $d_z = 8$ as the fiducial autoencoder and report the prediction performance in the next section.

Table 1
Table Summarizing the Performance Scores of Explosion Outcome Prediction Using Different Feature Sets

Features	Accuracy	Precision	Recall	F1 Score
$\langle x_m \rangle$, σ_x	0.68 ± 0.12	0.68 ± 0.13	0.69 ± 0.13	0.67 ± 0.12
$\xi_{1.75}$	0.83 ± 0.08	0.84 ± 0.06	0.86 ± 0.06	0.83 ± 0.07
μ_4, $M_4\mu_4$	0.70 ± 0.09	0.69 ± 0.08	0.69 ± 0.08	0.69 ± 0.08
$M_{\rm SiO}$, $\Delta\rho_{\rm SiO}$	0.89 ± 0.10	0.89 ± 0.09	0.91 ± 0.08	0.89 ± 0.10
Autoencoder ($d_z = 2$)	0.77 ± 0.07	0.79 ± 0.07	0.75 ± 0.07	0.74 ± 0.07
Autoencoder ($d_z = 8$)	0.84 ± 0.06	0.84 ± 0.07	0.83 ± 0.06	0.83 ± 0.06

Note. Each row corresponds to a different selection of feature parameter(s) used in the training and evaluation of the classifiers. Errors shown are the standard deviations of the respective scores

3.2. Explosion Outcome Prediction

We choose four sets of physics-based features for the explosion outcome prediction task, as listed in the left column of Table 1. The first feature set ($\langle x_m \rangle$, σ_x) are the mean and standard deviation of the truncated logarithmic density profiles, representing the most basic summary statistics of the density

profiles. The remaining three sets correspond to the physics-based features described in Section 2.2.

We train and evaluate a series of RF classifiers using our labeled data set. Performance scores are listed in Table 1 with the errors denoting the standard deviations over the five cross-validation splits. We find that our Si/O interface parameter set provides the best prediction of explosion outcomes, with a prediction accuracy of 0.89. By comparison, $\xi_{1.75}$ yields an accuracy of 0.83 and the Ertl condition 0.70. Due to the limited size of the labeled set, there are ≈10% fluctuations in the performance scores between different cross-validation splits as well as among repetitions of the RF training (from the randomness in tree building). However, the general trend of performance with different feature sets is robust.

The embedding feature vector extracted by the fiducial autoencoder gives a prediction accuracy of 0.84, comparable to both the Si/O interface and the compactness parameter. Even with a reduced embedding dimension of $d_z = 2$, the autoencoder feature vector still outperforms the Ertl parameters. It highlights that features obtained from the density profiles via an unsupervised, physics-agnostic manner can offer classification performance competitive with physics-based features. To examine whether retaining the absolute scale of density may impact prediction performance, we have trained a control classifier with mean densities appended to the autoencoder features and found negligible difference.

Identifying a prominent Si/O interface can be difficult, particularly with low-mass models, perhaps explaining some of the misclassifications. We find that stars between 12 and 15 $M_\odot$ lack prominent Si/O interfaces and tend to be more difficult to explode. According to Sukhbold et al. (2018), stars in this range may have multiple smaller, fragmented interfaces as well. We emphasize that our conclusion depends sensitively on both the progenitor profile and the neutrino microphysics included (see, for instance, Burrows et al. 2019). Regardless, multiple studies have found that models generally within this mass range are less likely to explode (Summa et al. 2016; O'Connor & Couch 2018b; Vartanyan et al. 2018; Burrows et al. 2019, 2019; Burrows & Vartanyan 2021; Wang et al. 2022).

The accuracy of the compactness parameter is surprising at first sight, given that studies have found no simple correlation between compactness and the explosion outcome (Burrows et al. 2020; Burrows & Vartanyan 2021). Indeed, we see no monotonic dependence of the explosion outcome on compactness in Figure 1. Rather, the RF classifier here identifies the nonlinear mapping between compactness and the explosion outcome from the training samples. Such a nonlinear dependence of the explosion outcome is suggestive of additional underlying physics that is not captured by the compactness parameter, perhaps in the nuances of the density profile.

Regardless of the metric used, we find predictive accuracy above 70%, indicating that all the metrics considered contain some physical information about the explosion outcome. The Ertl parameter just underperforms compared to simply using only the density and its standard deviation. While both the Ertl parameter and compactness contain information about the density of the progenitor, the former may obfuscate it through analytical complication, while the latter, which performs better, is oversimplified. Categorizing the density profile through prominent interfaces seems to be the best approach thus far to predicting explosion outcomes.

3.3. Utility of Label Propagation

Table 2 summarizes the results of the label propagation study. Unsurprisingly, classifiers trained with fewer labeled samples tend to have poorer prediction performance across feature sets. Even with 75% of the labeled training samples dropped, i.e., only with 20 training samples, the classifiers can still preserve reasonable prediction accuracy scores of about 0.6–0.8 (the "no LP" columns). This suggests that the feature spaces we investigated can be sampled reasonably well with about 20–40 models, and that most of the misclassifications reside in the overlaps of branches that may be resolved by additional training samples.

Label propagation offers only marginal improvements of a few percentage points across different feature sets. In some cases, e.g., with $\xi_{1.75}$ and autoencoder features of embedding dimension $d_z = 2$, label propagation can even diminish prediction performance. Such reduction in accuracy can be understood again by the complex discontinuities in the feature spaces and the stochasticity in modeling. With a small number of training samples, the classifiers can sometimes be misled by a single training sample to misclassify large parts of the feature space. With a much larger data set, the overlapping outcome branches will be better distinguished. We expect label propagation to be more effective in improving prediction accuracy with feature spaces that are smoother and less susceptible to model stochasticity. The unsupervised approach of feature extraction holds promise in uncovering such feature spaces.

4. Conclusions

We explored the utility of a machine-learning framework in predicting the explosion outcomes of massive stars based on their 1D progenitor models. We trained and evaluated a basic RF classifier as an explosion predictor using both physics-based and physics-agnostic features. In particular, we investigated the commonly used compactness parameter and Ertl conditions, a new feature set that quantifies the location and the extent of the density drop at the silicon/oxygen interface, and autoencoder features generated from the progenitor density profiles in an unsupervised manner. Applied to a set of 100 2D radiation hydrodynamical FORNAX simulations, we found that the new silicon/oxygen interface feature set has the best predictive power, with an accuracy of ≈90%, outperforming the compactness and the Ertl condition. More importantly, using the physics-agnostic autoencoder features, we obtained a predictive accuracy of ≈84%, second only to the silicon/oxygen interface features.

The competitive predictive performance of the autoencoder features revealed that the density profiles alone contain meaningful information about the explodability of the stellar progenitors. It suggests that exploration of the clusters and branches in the reduced-dimension embedding space holds promise in uncovering the underlying progenitor properties that foreshadow explosions. With more multidimensional explosion simulations in the near future, we expect the unsupervised approach to representing progenitor models to be profitable in the task of identifying more robust explosion physics.

5. Caveats and Future Prospects

The conclusions cited here assume neutrino heating as the dominant explosion mechanism, as is well understood to be the

Table 2
Table Comparing the Accuracy Scores of Different Feature Sets with 50% or 75% of the Labels Dropped in the RF Classifier Training Set

Features	Fully Supervised	50% Dropped		75% Dropped	
		No LP	LP	No LP	LP
$\langle x_m \rangle$, σ_x	0.68 ± 0.12	0.65 ± 0.14	0.71 ± 0.14	0.61 ± 0.06	0.63 ± 0.08
$\xi_{1.75}$	0.83 ± 0.08	0.81 ± 0.07	0.78 ± 0.07	0.77 ± 0.10	0.74 ± 0.12
μ_4, $M_4\mu_4$	0.70 ± 0.09	0.68 ± 0.11	0.67 ± 0.08	0.62 ± 0.08	0.65 ± 0.04
$M_{\rm SiO}$, $\Delta\rho_{\rm SiO}$	0.89 ± 0.10	0.84 ± 0.07	0.89 ± 0.09	0.82 ± 0.07	0.84 ± 0.12
Autoencoder ($d_z = 2$)	0.77 ± 0.07	0.76 ± 0.04	0.72 ± 0.07	0.71 ± 0.09	0.69 ± 0.06
Autoencoder ($d_z = 8$)	0.84 ± 0.06	0.74 ± 0.15	0.83 ± 0.09	0.71 ± 0.10	0.73 ± 0.06

Note. The "Fully Supervised" column corresponds to the accuracy using the full labeled training sets (Table 1). The "no LP" columns list the prediction performance trained directly from the reduced-size training set, while the "LP" columns show the performance with label propagation applied to the samples with the labels dropped.

case for the majority of garden-variety CCSNe. Intrinsically, the explosion outcome depends on various physical uncertainties such as microphysics, progenitor structure, convection, and nuclear burning rates. At the simulation level, a confluence of factors may also come into play, e.g., details of the code, simulation dimensions, and stochasticity. To properly characterize the explosion outcomes of populations of massive stars, careful cross-checking with different sets of supernova simulations will be required. Going beyond a binary depiction of explosion outcomes, future studies may start charting the explosion probability in different feature spaces using ensembles of simulations. Our model suite can be extended to include additional physics, including magnetorotational effects, for instance, to capture more of the physical parameter space yet unexplored in CCSNe simulations.

Importantly, the stellar progenitors used were all spherically symmetric, 1D models. Only recently (Müller et al. 2016b; Yoshida et al. 2019; Zha et al. 2019; Fields & Couch 2020, 2021; Fields 2022) have multidimensional progenitor models become available for simulation (Müller et al. 2018, 2019; Vartanyan et al. 2022; Zha et al. 2022). CCSNe simulations are sensitive to the ambient perturbations in the progenitor model (see, e.g., Burrows et al. 2019). The structure of the prominent compositional interfaces, and hence the explosion outcome, morphology, and nucleosynthetic yields, will differ between 3D and 1D progenitor models.

The relevant physical parameter space for the explosion outcome is both very large and poorly constrained. Even the presence of a strong interface is difficult to resolve, and sometimes absent, in many progenitors. Additional constraints, perhaps involving the Helium core mass or some other characterization of the density profile (Sukhbold et al. 2018; Wang et al. 2022) is needed to break the degeneracy in predicting explosion outcomes. We focused exclusively on the density profile when computing both the physics-based and physics-agnostic autoencoder features, but we could expand the feature sets to include also the temperature, electron-fraction, and/or entropy profiles, etc. from the progenitor stellar models. For example, multiple profiles can be readily incorporated as different input channels in the autoencoder architecture. Our main goal is to demonstrate the usefulness of the unsupervised feature extraction approach. We therefore did not conduct a thorough hyperparameter study for the autoencoder architecture. Exploring the utility of transformers (Vaswani et al. 2017), another neural network architecture that is effective in representing sequential data, will also be a promising future direction.

Furthermore, our data set was limited in size. We selected from approximately 1500 progenitor models and trained on 100 axisymmetric 2D simulations. Although the explosion outcome does not seem to differ greatly between 2D and 3D simulations (Vartanyan et al. 2019; Burrows & Vartanyan 2021), a significantly larger catalog of simulations, even in axisymmetry, would better populate the distribution of density profiles by progenitor, perhaps better resolving the clustering and outcome branches (Müller et al. 2016a; Sukhbold et al. 2018) seen in the different phase spaces explored here. At the very least, we would need tenfold more simulations (thousands), even in 2D, to have a more balanced and comprehensive data set. Yet even with the limited data set and our simple approach in both identifying a physical criterion of interest and apposite ML techniques, we were able to obtain promising results.

Machine-learning applications are not limited to a simple binary determination of explosion outcomes. Regression models can enable the prediction of explosion diagnostics, such as the energy and ejecta composition, for a given physical setup and progenitor model. Inverse modeling can facilitate the reconstruction of progenitor properties from observables. Data transformation and image segmentation, which have already seen some use categorizing observations, can be used to characterize morphological features of supernova remnants, such as nickel bullets and voids/clustering in the ejecta, which, jointly with inverse modeling, can characterize the structure of the stellar progenitor and its evolutionary history. Machine-learning techniques provide an invaluable perspective from which to map the initial stellar mass function to the distribution of residues, i.e., black holes and neutron stars (partitioned between failed and successful supernovae), and are exquisitely suitable for the upcoming era of all-sky surveys. The effort here presents a first step in this direction.

We are grateful to Daniel Kasen, Tianshu Wang, and Matthew Coleman for valuable insights and discussion. This research was funded by the Gordon and Betty Moore Foundation through grant GBMF5076, and by NASA awards ATP-80NSSC18K0560 and ATP-80NSSC22K0725. This research was supported in part by the National Science Foundation under grant No. NSF PHY-1748958. We acknowledge support from the U.S. Department of Energy Office of Science and the Office of Advanced Scientific Computing Research via the Scientific Discovery through Advanced Computing (SciDAC4) program and grant DE-SC0018297 (subaward 00009650) and support from the U.S. NSF under Grants AST-1714267 and PHY-1804048 (the latter via the

Max-Planck/Princeton Center (MPPC) for Plasma Physics). A generous award of computer time was provided by the INCITE program. That research used resources of the Argonne Leadership Computing Facility, which is a DOE Office of Science User Facility supported under Contract DE-AC02-06CH11357. We are also grateful for our computational resources through the Texas Advanced Computing Center (TACC) at The University of Texas at Austin via Frontera Large-Scale Community Partnerships under grant SC0018297 as well as the Leadership Resource Allocation under grant No. 1804048. In addition, this overall research project was part of the Blue Waters sustained-petascale computing project, which is supported by the National Science Foundation (awards OCI-0725070 and ACI-1238993) and the state of Illinois. Blue Waters was a joint effort of the University of Illinois at Urbana-Champaign and its National Center for Supercomputing Applications. This general project was also part of the "Three-Dimensional Simulations of Core-Collapse Supernovae" PRAC allocation support by the National Science Foundation (under award #OAC-1809073). Moreover, we acknowledge access under the local award #TG-AST170045 to the resource Stampede2 in the Extreme Science and Engineering Discovery Environment (XSEDE), which is supported by National Science Foundation grant No. ACI-1548562. Finally, the authors employed computational resources provided by the TIGRESS high performance computer center at Princeton University, which is jointly supported by the Princeton Institute for Computational Science and Engineering (PICSciE) and the Princeton University Office of Information Technology, and acknowledge our continuing allocation at the National Energy Research Scientific Computing Center (NERSC), which is supported by the Office of Science of the US Department of Energy (DOE) under contract DE-AC03-76SF00098. Use was made of computational facilities purchased with funds from the National Science Foundation (CNS-1725797) and administered by the Center for Scientific Computing (CSC). The CSC is supported by the California NanoSystems Institute and the Materials Research Science and Engineering Center (MRSEC; NSF DMR 1720256) at UC Santa Barbara.

Software: KEPLER (Sukhbold et al. 2016), FORNAX (Skinner et al. 2019), Jupyter (Kluyver et al. 2016), Matplotlib (Hunter 2007), NumPy (Oliphant 2006), PyTorch (Paszke et al. 2019), SKLEARN (Pedregosa et al. 2011).

Data Availability

The data underlying this article will be shared on reasonable request to the corresponding author.

ORCID iDs

Benny T.-H. Tsang https://orcid.org/0000-0002-6543-2993
David Vartanyan https://orcid.org/0000-0003-1938-9282
Adam Burrows https://orcid.org/0000-0002-3099-5024

References

Aniyan, A. K., & Thorat, K. 2017, ApJS, 230, 20
Ball, N. M., & Brunner, R. J. 2010, IJMPD, 19, 1049
Bengyat, O., & Gal-Yam, A. 2022, ApJ, 930, 31
Bethe, H. A., & Wilson, J. R. 1985, ApJ, 295, 14
Boccioli, L., Roberti, L., Limongi, M., Mathews, G. J., & Chieffi, A. 2022, arXiv:2207.08361
Burrows, A., & Goshy, J. 1993, ApJL, 416, L75
Burrows, A., Radice, D., & Vartanyan, D. 2019, MNRAS, 485, 3153
Burrows, A., Radice, D., Vartanyan, D., et al. 2020, MNRAS, 491, 2715
Burrows, A., & Vartanyan, D. 2021, Natur, 589, 29
Burrows, A., Vartanyan, D., Dolence, J. C., Skinner, M. A., & Radice, D. 2018, SSRv, 214, 33
Caldeira, J., Wu, W. L. K., Nord, B., et al. 2019, A&C, 28, 100307
Cheng, T.-Y., Li, N., Conselice, C. J., et al. 2020, MNRAS, 494, 3750
Colgate, S. A., & White, R. H. 1966, ApJ, 143, 626
Ivezić, Z. ˇ, Connolly, A. J., VanderPlas, J. T., & Gray, A. 2014, Statistics, Data Mining, and Machine Learning in Astronomy: A Practical Python Guide for the Analysis of Survey Data (Princeton, NJ: Princeton Univ. Press)
Couch, S. M., Warren, M. L., & O'Connor, E. P. 2020, ApJ, 890, 127
Curtis, S., Wolfe, N., Fröhlich, C., et al. 2021, ApJ, 921, 143
Dolence, J. C., Burrows, A., & Zhang, W. 2015, ApJ, 800, 10
Ertl, T., Janka, H. T., Woosley, S. E., Sukhbold, T., & Ugliano, M. 2016, ApJ, 818, 124
Fields, C. E. 2022, ApJL, 924, L15
Fields, C. E., & Couch, S. M. 2020, ApJ, 901, 33
Fields, C. E., & Couch, S. M. 2021, ApJ, 921, 28
Fluke, C. J., & Jacobs, C. 2020, WIRES DATA MIN KNOWL, 10, e1349
Fryer, C. L. 1999, ApJ, 522, 413
Gheller, C., & Vazza, F. 2022, MNRAS, 509, 990
Glas, R., Just, O., Janka, H. T., & Obergaulinger, M. 2019, ApJ, 873, 45
Gogilashvili, M., & Murphy, J. W. 2022, MNRAS, 515, 1610
Harada, A., Nishikawa, S., & Yamada, S. 2022, ApJ, 925, 117
He, K., Zhang, X., Ren, S., & Sun, J. 2015, arXiv:1502.01852
Horowitz, B., Dornfest, M., Lukić, Z., & Harrington, P. 2021, arXiv:2106.12675
Hunter, J. D. 2007, CSE, 9, 90
Ishida, E. E. O., Kornilov, M. V., Malanchev, K. L., et al. 2021, A&A, 650, A195
Karpov, P. I., Huang, C., Sitdikov, I., et al. 2022, arXiv:2205.08663
Kingma, D. P., & Ba, J. 2014, arXiv:1412.6980
Kluyver, T., Ragan-Kelley, B., Pérez, F., et al. 2016, in Positioning and Power in Academic Publishing: Players, Agents and Agendas, ed. F. Loizides & B. Schmidt (Amsterdam: IOS Press), 87
Krastev, P. G. 2022, Galax, 10, 16
Ksoll, V. F., Ardizzone, L., Klessen, R., et al. 2020, MNRAS, 499, 5447
Kuroda, T., Arcones, A., Takiwaki, T., & Kotake, K. 2020, ApJ, 896, 102
Kuroda, T., Fischer, T., Takiwaki, T., & Kotake, K. 2022, ApJ, 924, 38
Laplace, E., Justham, S., Renzo, M., et al. 2021, A&A, 656, A58
Mabanta, Q. A., Murphy, J. W., & Dolence, J. C. 2019, ApJ, 887, 43
Malanchev, K. L., Pruzhinskaya, M. V., Korolev, V. S., et al. 2021, MNRAS, 502, 5147
Marek, A., Dimmelmeier, H., Janka, H.-T., Müller, E., & Buras, R. 2006, A&A, 445, 273
Müller, B., Gay, D. W., Heger, A., Tauris, T. M., & Sim, S. A. 2018, MNRAS, 479, 3675
Müller, B., Heger, A., Liptai, D., & Cameron, J. B. 2016a, MNRAS, 460, 742
Müller, B., Tauris, T. M., Heger, A., et al. 2019, MNRAS, 484, 3307
Müller, B., Viallet, M., Heger, A., & Janka, H.-T. 2016b, ApJ, 833, 124
Mustafa, M., Bard, D., Bhimji, W., et al. 2019, ComAC, 6, 1
Nagakura, H., Burrows, A., Radice, D., & Vartanyan, D. 2019, MNRAS, 490, 4622
Naul, B., Bloom, J. S., Pérez, F., & van der Walt, S. 2018, NatAs, 2, 151
Obergaulinger, M., & Aloy, M. Á. 2021, MNRAS, 503, 4942
O'Connor, E., & Ott, C. D. 2011, ApJ, 730, 70
O'Connor, E., & Ott, C. D. 2013, ApJ, 762, 126
O'Connor, E. P., & Couch, S. M. 2018a, ApJ, 865, 81
O'Connor, E. P., & Couch, S. M. 2018b, ApJ, 854, 63
Oliphant, T. E. 2006, A guide to NumPy, Vol. 1 (USA: Trelgol Publishing)
Ott, C. D., Roberts, L. F., da Silva Schneider, A., et al. 2018, ApJL, 855, L3
Paszke, A., Gross, S., Massa, F., et al. 2019, in Advances in Neural Information Processing Systems 32, ed. H. Wallach et al. (New York: Curran Associates Inc.), 8024
Pedregosa, F., Varoquaux, G., Gramfort, A., et al. 2011, J Mach Learn Res., 12, 2825
Pejcha, O., & Thompson, T. A. 2012, ApJ, 746, 106
Perego, A., Hempel, M., Fröhlich, C., et al. 2015, ApJ, 806, 275
Portillo, S. K. N., Parejko, J. K., Vergara, J. R., & Connolly, A. J. 2020, AJ, 160, 45
Radice, D., Burrows, A., Vartanyan, D., Skinner, M. A., & Dolence, J. C. 2017, ApJ, 850, 43
Raives, M. J., Couch, S. M., Greco, J. P., Pejcha, O., & Thompson, T. A. 2018, MNRAS, 481, 3293

Skinner, M. A., Dolence, J. C., Burrows, A., Radice, D., & Vartanyan, D. 2019, ApJS, 241, 7
Sukhbold, T., Ertl, T., Woosley, S. E., Brown, J. M., & Janka, H.-T. 2016, ApJ, 821, 38
Sukhbold, T., Woosley, S. E., & Heger, A. 2018, ApJ, 860, 93
Summa, A., Hanke, F., Janka, H.-T., et al. 2016, ApJ, 825, 6
Summa, A., Janka, H.-T., Melson, T., & Marek, A. 2018, ApJ, 852, 28
Tsang, B. T. H., & Schultz, W. C. 2019, ApJL, 877, L14
Ugliano, M., Janka, H.-T., Marek, A., & Arcones, A. 2012, ApJ, 757, 69
van Roestel, J., Duev, D. A., Mahabal, A. A., et al. 2021, AJ, 161, 267
Vartanyan, D., Burrows, A., Radice, D., Skinner, M. A., & Dolence, J. 2018, MNRAS, 477, 3091
Vartanyan, D., Burrows, A., Radice, D., Skinner, M. A., & Dolence, J. 2019, MNRAS, 482, 351
Vartanyan, D., Coleman, M. S. B., & Burrows, A. 2022, MNRAS, 510, 4689
Vartanyan, D., Laplace, E., Renzo, M., et al. 2021, ApJL, 916, L5
Vaswani, A., Shazeer, N., Parmar, N., et al. 2017, arXiv:1706.03762
Vaytet, N. M. H., Audit, E., Dubroca, B., & Delahaye, F. 2011, JQSRT, 112, 1323
Villanueva-Domingo, P., & Villaescusa-Navarro, F. 2022, arXiv:2204.13713
Villanueva-Domingo, P., Villaescusa-Navarro, F., Anglés-Alcázar, D., et al. 2022, ApJ, 935, 30
Villar, V. A., Cranmer, M., Contardo, G., Ho, S., & Yao-Yu Lin, J. 2020, arXiv:2010.11194
Vogl, C., Kerzendorf, W. E., Sim, S. A., et al. 2020, A&A, 633, A88
Wang, T., Vartanyan, D., Burrows, A., & Coleman, M. S. B. 2022, arXiv:2207.02231
Williamson, M., Modjaz, M., & Bianco, F. B. 2019, ApJL, 880, L22
Yoshida, T., Takiwaki, T., Kotake, K., et al. 2019, ApJ, 881, 16
Zha, S., Leung, S.-C., Suzuki, T., & Nomoto, K. 2019, ApJ, 886, 22
Zha, S., O'Connor, E. P., Couch, S. M., Leung, S.-C., & Nomoto, K. 2022, MNRAS, 513, 1317