

---

# NEURAL NETWORKS OPTIMIZED BY GENETIC ALGORITHMS IN COSMOLOGY

---

**Isidro Gómez-Vargas**

Instituto de Ciencias Físicas,  
Universidad Nacional Autónoma de México,  
62210, Cuernavaca, Morelos, México.  
igomez@icf.unam.mx

**Joshua Briones Andrade**

Facultad de Ciencias,  
Universidad Nacional Autónoma de México,  
Ciudad de México, México.  
joshuabriones@ciencias.unam.mx

**J. Alberto Vázquez**

Instituto de Ciencias Físicas,  
Universidad Nacional Autónoma de México,  
62210, Cuernavaca, Morelos, México.  
javazquez@icf.unam.mx

September 7, 2022

## ABSTRACT

The applications of artificial neural networks in the cosmological field have shone successfully during the past decade, this is due to their great ability of modeling large amounts of datasets and complex nonlinear functions. However, in some cases, their use still remains controversial because their ease of producing inaccurate results when the hyperparameters are not carefully selected. In this paper, to find the optimal combination of hyperparameters that describe the artificial neural networks, we propose to take advantage of the genetic algorithms. As a proof of the concept, we analyze three different cosmological cases to test the performance of the new architecture achieved with the genetic algorithms and compare the output with the standard process, consisting of a grid with all possible configurations. First, we carry out a model-independent reconstruction of the distance modulus using a Type Ia Supernovae compilation. Second, the neural networks learn to solve dynamical system of the Universe's content, and finally with the latest Sloan Digital Sky Survey data release we train the networks for the classification of astronomical objects. We found that the genetic algorithms improve considerably the generation of the architecture, which can ensure more confidence in their physical results because of the better performance in the metrics with respect to the grid method.

**Keywords** Observational cosmology · Artificial Neural Networks · Genetic Algorithms · Hyperparameter tuning

## 1 Introduction

Throughout cosmology there exists a variety of numerical and statistical techniques that allow the study of complex theoretical models and to process large amounts of observational measurements. Despite a growing amount of observational evidence, there are still several cosmological tensions with unknown physical explanations, this encourages the search for new analysis tools to extract valuable information from the data and to use the computational resources more efficiently. During the last decades, Machine Learning has offered effective alternatives incorporated into the data analysis field, in particular Deep Learning. For instance, the Artificial Neural Networks (ANNs) have been used with great performance in various tasks such as regression, classification, image processing and time series, among others [1]. In cosmology, ANNs have recently been considered in several applications, such as, model-independent reconstructions (nonlinear regression) [2, 3, 4, 5, 6], to speed up numerical calculations [7, 8, 9, 10, 11] and on the classification of different objects [12, 13, 14, 11]. However, most of these works built the network architecture by generating a grid with

a large amount of hyperparameter values to select the best one, within the possible combinations, and in other scenarios just on an empirical way, which may lead to inaccurate results.

Despite their great advantages, ANNs have two drawbacks. First, the fact that the networks have thousands or millions of parameters (called weights) generates a hard interpretation of them. Second, ANNs have also several hyperparameters that must be picked out (e.g. number of layers, nodes, activation function, batch size, etc.) in order to have acceptable predictions and therefore the results depend on their selection. Despite that several combinations are able to generate a good neural network model, there are even more bad combinations that will achieve incorrect predictions and, in cosmology, a spurious and imprecise physical interpretation. If the hyperparameters configuration is adequate, it implies a good balance between the bias and variance of the neural model, that is, the neural network is neither over-fitted nor under-fitted [15], therefore the predictions should be reliable and thus underestimate the weak interpretation of the multiple weights.

There are several strategies for finding the appropriate values of the hyperparameters in a neural network [16, 17, 18, 19]. The most standard approach is to propose a multidimensional grid of possible values for the hyperparameters [16], evaluate all possible combinations and determine, by comparison, which has the best performance. Newer approaches are based in mathematical optimization or metaheuristic algorithms, both of which consist in specialized algorithms to find the optimal value for a given function. Therefore, the search for the best combination of hyperparameters of neural networks is posed as an optimization problem.

Genetic algorithms, by themselves, have been investigated in cosmology for quasar parameterisation [20], nonparametric reconstructions [21, 22], string theory analysis [23], to name just a few. In other research areas, there are various cases in which artificial neural networks and genetic algorithms have been applied together; for example, in medicine [24, 25, 26], seismology [27] and in string theory [28], among others.

In several fields it is still uncommon to pay particular attention to the hyperparameter tuning during the construction of the neural network model. Regularly, a usual strategy is the hyperparameter grid and in several cases it may be a good enough option, however there are better ways to optimize this selection. In this paper, for this task, we explore the use of the Genetic Algorithms (GA), the most popular metaheuristic algorithm, and we compare their performance with the standard grid of hyperparameters. In this context, the goal of the genetic algorithm is to find the best combination of neural network hyperparameters that minimizes the target function.

This paper is structured as follows. In the next section we provide a brief overview about neural networks, genetic algorithms and the hyperparameter tuning approaches. In Section 3 we explain the technical details about our implementation. Section 4 has three cosmological study cases, first we made a model-independent reconstruction of the distance modulus using a Type Ia Supernovae compilation; secondly, training neural network models using the analytical results of the dynamical system of the matter density of the Universe; thirdly, a classification of stellar objects using information from the Sloan Digital Sky Survey Data Release 17. Finally, in Section 5 we describe our final reflections about this research.

## 2 Machine learning background

Machine learning is the field of Artificial Intelligence focused on the mathematical modeling of the data, it extracts the intrinsic properties of datasets by minimizing an objective function through many iterations until an acceptable combination of model parameters is found. In recent years, the most successful types of machine learning models are artificial neural networks, which have thousands or millions of parameters, called weights, that allow the modeling of any nonlinear function [29].

Finding the correct hyperparameters, or in other words the best neural network architecture, is a hard task. The classic method for this is to generate several combinations of hyperparameters and evaluate all of them until the best one is found. In recent years, taking advantage of the existing computing power, various optimization and parameter estimation techniques have been applied to find these hyperparameters more efficiently. In particular, the metaheuristic optimization algorithms (i.e. the genetic algorithm), which allow finding the best solution to an optimization problem without using derivatives.

In this section we describe very briefly artificial neural networks, genetic algorithms and the hyperparameter tuning.

### 2.1 Artificial neural networks

Artificial neural networks have been applied in various scientific fields because of their ability to model large and complex datasets. The universal approximation theorem guarantees that ANNs can model any nonlinear function [29], making them a powerful tool in modeling datasets where the intrinsic relationships of their variables are unknown and

most of the time multidimensional. A complete review of neural networks is beyond the scope of this article; there are great references in the literature to delve into this topic in a formal way [1, 30, 31]; for a basic introduction, their main algorithms and cosmological context, see [11].

Inspired by nature, an artificial neural network (ANN) consists of a computational model that aims to emulate the synapse through interconnected layers of units called neurons or nodes, which make up its basic information processing elements. In the simplest type of neural network, the feedforward neural network, there are three types of layers: an input layer that receives the input, hidden layers responsible for extracting patterns and producing nonlinearity effects, and finally the output layer that presents the results of the prediction.

The intrinsic parameters of ANNs are known as hyperparameters, which, unlike the weights, are not adjusted during the training and must be configured in advanced. Examples of hyperparameters are the number of layers, the number of nodes per layer, the number of epochs and the activation function. In addition, since ANNs use a gradient descent algorithm and a backpropagation algorithm [32], the parameters of these algorithms can also be hyperparameters of the ANN model, e.g., batch size and learning rate, among others. In practice, some hyperparameters are fixed and others remain as free parameters that are founded by a tuning strategy.

## 2.2 Genetic algorithms

Genetic algorithms are inspired by genetic populations which consider any possible solution of an optimization problem as an individual. It is beyond the scope of this article a full background of genetic algorithms, but for interested readers we recommend the following references [33, 34, 35].

Inspired by genetic populations, the first step of a genetic algorithm is to generate several individuals, within the search space, and define them as a population. Then, through operations such as descent, crossover and mutation, over several iterations (called generations) the population gets closer to the optimum of an target function. Genetic algorithms are popular for their ability to solve large-scale nonlinear and nonconvex optimization problems [36] and for difficult search situations [37]. In a problem approached with a genetic algorithm, it is necessary to choose the objective, or target, function to optimize, define the search space and select the genetic parameters: crossover, mutation and elitism (there may be others, but these are the most common). It is necessary to assign probability values to the crossover and mutation operators and for each iteration two individuals can have a crossover or a single individual a mutation with these probabilities. In addition, a selection operator is introduced, which defines how some individuals can survive to the next generation. The value of elitism indicates how many individuals are bound to pass to the next generation, so it is a positive integer value. Thus, in a few words the genetic algorithm works as follow: it generates an initial population within the search space and, generation by generation, the individuals are modified by the operators and the evaluation of the objective function, while approaching the optimum of the target function.

## 2.3 Neural Networks hyperparameter tuning

In this paper we focus on two approaches to tuning the hyperparameters of neural networks. First, the classic grid of hyperparameters and second, the use of genetic algorithms to find a good combination of hyperparameters. Here are some highlights of both.

### 2.3.1 Conventional grid

The typical approach to finding a correct combination of hyperparameters in the ANN is to go through an array of possible hyperparameter values and evaluate each combination to choose the best that minimizes the neural network loss function [16]. This involves training as many times as there are combinations, and is very computationally expensive. Another technique that attempts to reduce this cost is random search, in which hyperparameter combinations are randomly sampled; however, it still has the same problems and depends on the size of the search space for its efficiency. In both approaches the best solution is always within the initial set of combinations and, however, a lot of configurations have to be evaluated.

### 2.3.2 Using genetic algorithms

The search for the hyperparameters of an ANN can be considered an optimization problem. Due to the increased number of hyperparameters, the search space is likely to be complex and high-dimensional. Classical optimization methods involving derivatives can be very difficult to implement in this kind of scenario, therefore genetic algorithms are a very interesting way to solve this problem.

The crucial step in using genetic algorithms in hyperparameter fitting is to define the fitness function, or target function. For the case of neural networks, the loss function can be used as fitness. The loss function during neural network training

aims to be minimized, therefore the task of genetic algorithms is to find the best combination of hyperparameters that minimizes the target function. Several research works, unrelated to cosmology, have already combined these two powerful tools and in most cases have promising results [38, 39, 40, 41].

### 3 Methodology

We use `tensorflow` [42] to program the ANN models and the DEAP library [43] to implement the genetic algorithms, both in Python programming language. We developed a Python library called NNOGADA<sup>1</sup> in which a simple genetic algorithm searches the best hyperparameters for a neural network.

In this framework, the objective function of the genetic algorithm must be some metric of the Artificial Neural Network; typically it would be the loss function, but for classification problems it could be accuracy, precision or something similar. In the case of the loss function, the problem would be a minimization and in the case of a classification metric, a maximization. In all cases in our study, we use the loss function and therefore have minimization problems.

The first step is to define the hyperparameters of the neural network model to be found. In practice, there are some hyperparameters that have values recommended by the literature or by experienced users on certain types of problems. In this work, as variable hyperparameters we choose the number of layers, the number of nodes, the learning rate and the batch size. With these we define the search space of the optimization problem. The gradient descent algorithm in our neural network models is Adam [44], and its hyperparameter that we tune is precisely the learning rate. Secondly, it is necessary to define the possible values for each hyperparameter, where the hyperparameter grid algorithm will search for the best combination, and those that the genetic algorithm will use to generate the first population.

In the case of the genetic algorithm, the possible values of the free hyperparameters must be encoded in a way that the genetic algorithm can understand them (binary, hexadecimal, etc.). Next, it is necessary to define the population size and the number of generations. Also the probability of mutation, crossover and elitism. Once this configuration of the genetic algorithm is established, we can use the neural model as a function to optimize. To have a fair comparison, in this work we chose these parameters of the genetic algorithms to have a number of neural network evaluations similar to the hyperparameter grid cases; the formal selection of the parameters of the genetic algorithm is out of the scope of this paper.

For the genetic algorithms, in all of the following case studies in Section 4, we set the tournament method [45] for selection with size 2, binary coding, using elitism with a hall of fame size equal to 1. Uniform crossover operator that modifies in place the two individuals in the sequence, with an independent probability of 0.5 for each attribute to exchange. As a mutation operator, a bit flip [46], with a probability of 0.1 for each attribute to flip. For technical details of the operators used, the related DEAP library reference can be consulted [47, 43, 48]. On the other hand, we varied and tested different values of crossover and mutation probabilities, population size and number of generations with the goal to analyze their effect in the performance of the genetic algorithms. Figure 1 summarizes the implementation of the hyperparameter tuning with genetic algorithms.

In all cases, we split the datasets into a training, validation and test sets. The first is used to adjust the weights during training, the validation set is used to evaluate the performance of each neural network model during training and, finally, the test set contains information not used in training but useful to measure the generalization capability of the neural networks. For the neural networks of the same example, we fix the number of epochs and report the lowest loss function for each case.

### 4 Cases of study

We have chosen three different cosmological problems to test hyperparameter tuning with genetic algorithms and in all of them we use the type of neural networks known as feedforward networks (also called multilayer perceptrons); however, the method can be easily implemented in other cosmological scenarios and with more complex neural network architectures such as convolutional or recurrent. In the following examples, we compare the performance of a hyperparameter grid and the genetic algorithms for finding an acceptable combination of hyperparameters in neural network models. We test these two strategies in three different cosmological contexts, similar to the problems presented in [11]. The Table 1 shows our choice of possible hyperparameter values for each of the following subsections, with which we construct the hyperparameter grids and the initial populations for the genetic algorithms.

---

<sup>1</sup>Neural Networks Optimized by Genetic Algorithms for Data Analysis: <https://github.com/igomezv/nnogada>

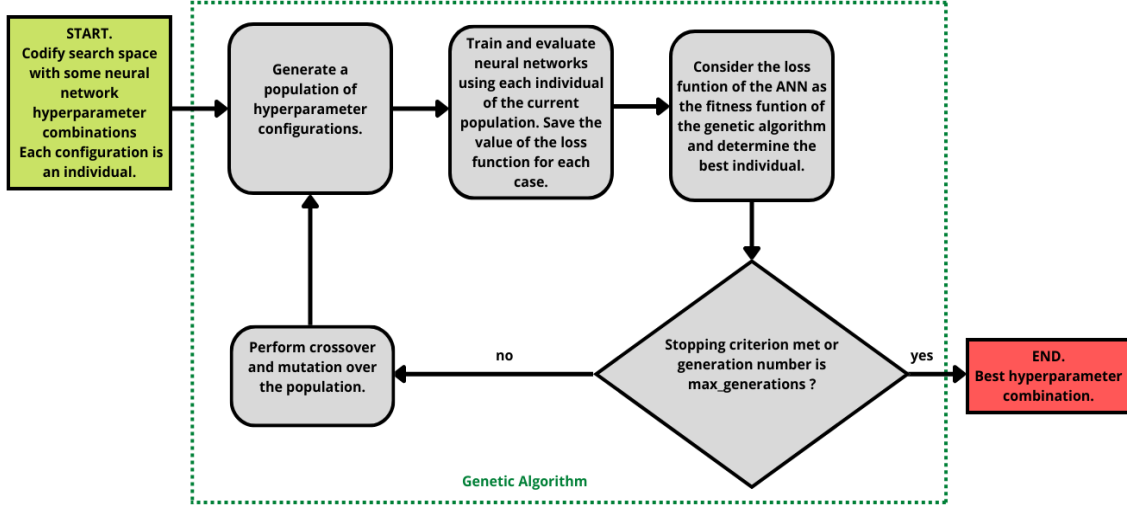


Figure 1: Diagram of neural network hyperparameter tuning with a genetic algorithm.

	Section 4.1	Section 4.2	Section 4.3
Number of layers	[1, 2, 3, 4]	[1, 2, 3, 4]	[3, 4]
Number of nodes	[50, 100, 150, 200]	[50, 100, 150, 200]	[100, 200]
Learning rate	[1e-4, 1e-3]	[1e-4, 1e-3]	[1e-5, 1e-4, 1e-3, 1e-2]
Batch size	[2, 4, 8, 16]	[8, 16]	[128, 256]

Table 1: Hyperparameters for the three cases of the Section 4.

#### 4.1 Reconstruction of distance modulus $\mu(z)$

Model-independent cosmological reconstructions refer to the use of statistical or computational techniques to generate a model of a cosmological observable without assuming an underlying theory. In cosmology, several techniques have been used such as histogram density estimators [49], Principal Component Analysis (PCA) [50, 51], smoothed step functions [52], gaussian processes [53, 54, 55, 56], extrapolation methods [57], Bayesian nodal free-form methods [58, 59], evolutionary algorithms [60, 22, 21] and recently, neural networks [2, 3, 6, 4].

In this example, we perform a reconstruction of the distance modulus, that already have been approached with this type of computational models [2, 3, 4], sometimes with a grid of hyperparameters for tuning the neural network architecture and others without any criterion about this.

We assume a spatially flat universe, for which the relationship between the luminosity distance  $d_L$  and the comoving distance  $D(z)$  is given by:

$$d_L(z) = \frac{1}{H_0}(1+z)D(z), \quad \text{with} \quad D(z) = H_0 \int \frac{dz}{H(z)}, \quad (1)$$

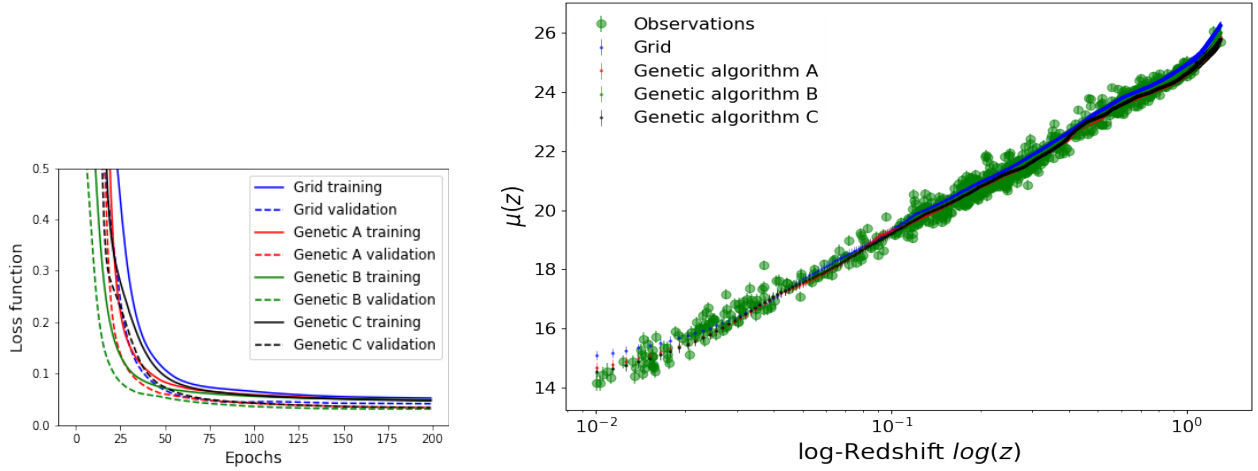
Thus, the observable quantity is computed by the distance modulus  $\mu(z) = 5 \log d_L(z) + 25$ . The SNeIa compilation used in this work corresponds to the Joint Lightcurve Analysis (JLA), with 740 Type Ia supernovae [61], it has as attributes the redshift of the measurement, the distance modulus and its statistical error; in addition, it has a covariance matrix with the systematic errors. We employ the diagonal of the covariance matrix, and with associate errors we add them to the statistical error. Then our neural network should have only one node in the input layer, corresponding to the redshift, and 2 nodes in the output layer for the distance modulus and the error (statistical plus systematic).

The hyperparameters have been searched, varying the number of layers, number of nodes, batch size and learning rate, for the values shown in the third column of the Table 1. Then, the grid of hyperparameters evaluates 128 different neural networks architectures to determine the best. For the genetic algorithms, as it can be seen in the Table 2, we set different values for the mutation probability, crossover probability, population size and number of generations.

In all cases models were trained along 200 epochs. In the Table 2 it can be noticed the result for the hyperparameter tuning using a grid and genetic algorithms with different parameters. For the test set, we use the mean squared error (MSE) as a metric to evaluate the performance of the neural network models. We can notice a little better performance

	Grid	Genetic A	Genetic B	Genetic C
Population size	-	5	5	10
Max generations	-	10	10	5
Crossover	-	0.5	0.5	0.2
Mutation	-	0.5	0.8	0.8
Hyperparameter results				
Layers	4	3	4	3
Nodes	200	100	100	100
Learning rate	0.0001	0.0001	0.0001	0.0001
Batch size	16	4	4	4
Metric				
MSE	0.0411	0.0334	0.0310	0.0337
Evaluations	128	32	42	40

Table 2: Results of neural nets training with JLA compilation.

Figure 2: *Left*: Loss function behavior for the training of the neural networks using the differential equations system. *Right*: Distance modulus reconstruction.

for the results of the genetic algorithms, even when it required less evaluations of neural networks architectures. Because, in this problem the numerical precision is very relevant, we can see in Figure 2 that indeed the genetic algorithms obtain better reconstructions for the distance modulus, particularly in the lower redshifts values.

Our results suggest that, in previous cosmological work [2, 3, 10, 6] in which function reconstructions are performed with neural networks, there is a possibility that a better architecture could be found using genetic algorithms instead of the use of the hyperparameter grid, and evidently for cases where no strategy is employed to find the correct architecture.

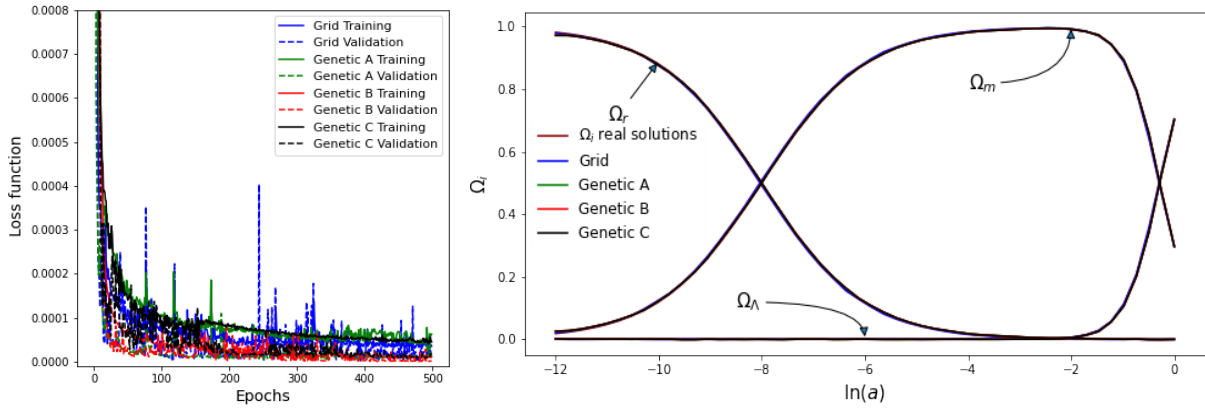
## 4.2 Cosmological differential equations

A system of differential equations can requires a lot of computational resources, especially when it must to be evaluated multiple times and artificial neural networks have been studied by solve this type of situations [62, 63]. In this case, we use a system of differential equations, recurrent in cosmology, which describes the evolution in time of the adimensional density parameters of the components of the Universe. We have trained the ANNs with functions dependent on the initial conditions of the differential equations. We consider a flat Universe whose evolution goes back to the early times when radiation dominates and we assume that its temperature is measured with great reliability, therefore we have a present radiation parameter  $\Omega_{r,0} = 10^{-4}$ . Then we vary the adimensional matter density  $\Omega_{m,0}$  and Hubble parameter  $H_0$  in the Equation 3 to have the initial conditions for Equation 2. With these parameters, the solution of the differential equations can be treated as a function  $\Omega_i(N, \Omega_{m,0}, H_0)$ .

$$\Omega'_i = 3(\Pi - \gamma_i)\Omega_i, \quad (2)$$

	Grid	Genetic A	Genetic B	Genetic C
Population size	-	11	10	10
Max generations	-	5	5	5
Crossover	-	0.8	0.5	0.2
Mutation	-	0.2	0.2	0.8
Hyperparameter results				
Layers	2	4	2	4
Nodes	150	150	200	150
Learning rate	0.001	0.001	0.0001	0.001
Batch size	16	8	2	16
Metric				
MSE	1.5437e-05	7.9833e-06	1.5655e-06	1.1097e-05
Evaluations	64	57	34	45

Table 3: Results of ANN training with cosmological differential equations.

Figure 3: *Left*: Loss function behavior for the training of the neural networks using the differential equations system. *Right*: Comparison of the analytical differential equations system and the predictions with the neural networks.

with  $\Omega_i$  the dimensionless density parameters,  $\Pi = \sum_i \gamma_i \Omega_i$ , where  $\gamma_i$  comes from the equation of state and describes each type of fluid (radiation  $\gamma_r = 4/3$ , baryonic and dark matter  $\gamma_m = 1$ , or dark energy as cosmological constant  $\gamma_\Lambda = 0$ ); the prime notation means derivative with respect to the e-fold parameter  $N = \ln(a)$ . The Friedmann equation becomes a constraint for the density parameters at all time:

$$\sum_i \Omega_i = 1. \quad (3)$$

The dataset was generated from defining intervals of the initial conditions:  $\Omega_{m,0} \in [0.1, 0.4]$ ,  $N = \ln(a) \in [-12, 0]$  and  $H_0 \in [65, 80]$ . Subsequently, a grid was created with all the combinations within these intervals, resulting in a training set of 30,750 samples for the training process. For more details about this procedure and its underlying cosmology, please refer to [11].

The neural network architecture must to have 3 nodes in the first layer related to  $\ln(a)$ ,  $\Omega_{m,0}$  and  $H_0$ , and 4 nodes in the output layer corresponding to  $\Omega_m(\ln(a))$ ,  $\Omega_r(\ln(a))$ ,  $\Omega_\Lambda(\ln(a))$  and  $H(\ln(a))$ .

Varying the hyperparameters shown in the Table 1 (batch size, learning rate, number of layers and number of nodes), the combinations founded with the grid method and the genetic algorithms are included in the Table 3.

All the neural network models were trained along 500 epochs. In the Table 3, we can see the results of the hyperparameter grid and genetic algorithms. We can notice that the neural networks obtained from the genetic algorithms search have a better performance in the MSE metric with fewer evaluations than those of the grid and regardless of different values of mutation and crossover probabilities. This example is the same as the one in Section 5 of the Reference [11], the difference is the hyperparameter tuning of the neural network. We can see that in the referred work, there are arbitrary values to all the hyperparameters involved and the MSE reaches a value  $2.9 \times 10^{-5}$ . In our case, for the MSE metric we

Attribute	Description
$\alpha$	right ascension
$\delta$	Declination
<b>u, g, r, i, z</b>	UV, green, red, near-infrared, far-infrared filters
<b>class</b>	Type of object: Stellar object classification ( <i>galaxy</i> , <i>star</i> or <i>quasar</i> )

Table 4: Attributes of the SDSS DR17 dataset

	Grid	Genetic A	Genetic B	Genetic C
Population size	-	8	8	5
Max generations	-	10	10	10
Crossover	-	0.2	0.5	0.5
Mutation	-	0.5	0.5	0.8
Hyperparameter results				
Layers	3	3	4	3
Nodes	100	200	200	200
Learning rate	0.001	0.001	0.001	0.001
Batch size	128	256	256	128
Metrics				
MSE	0.3041	0.3066	0.29869	0.28966
Accuracy	0.8836	0.8865	0.8872	0.8903
Precision	0.8816	0.8859	0.88608	0.88929
F1	0.8817	0.8858	0.8863	0.888747
Evaluations	32	49	54	41

Table 5: Results of ANN training with the SDSS data.

obtain one order of magnitude less in several cases, with a slightly better value for the genetic algorithms results, and we can be sure that the neural network models are better than a considerably number of hyperparameter combinations and as we can see in Figure 3, they have a very well performance.

### 4.3 Classification of cosmological objects

Classification is a common task in astronomy and cosmology, and neural networks have been very successful in tackling these type of problems [64, 65, 66, 67]. In this case we classify the type of stellar objects given a set of measurement features from the Sloan Digital Sky Survey Data Release 17 (SDSS-DR17) [68] which contains 100,000 observations. Each observation has 18 different attributes; however we only consider the 7 shown in the Table 4. In addition, each record has an associate class that indicates the type of stellar object: galaxy, star or quasar.

In the Table 4, the right ascension and declination are the celestial coordinates. The attributes u, g, r, i, z indicate information about the optical filters used in the measurements. Because these astronomical objects are related to the distance from the Earth, we remove the redshift attribute to avoid this information and not facilitate the training of the neural network.

In this case, we vary three hyperparameters: number of layers, number of nodes by layer (the same for all the layers) and the learning rate, such it can be noticed in the Table 1. The number of epochs is 50. The loss function fixed is categorical crossentropy, rectifier linear unit (ReLU) as activation function in the hidden layers and SoftMax in the last one.

For the above mentioned, the neural network model have 7 nodes in the input layer (the attributes of the Table 4) and 3 nodes in the output layer, one node by class. Using the hyperparameter grid varying the number of layers, number of nodes, batch size and learning rate (with the values shown in Table 1), we have a total of 32 combinations. And we choose three different configurations for the genetic algorithms regarding to the crossover and mutation probabilities, population size and generations.

The Table 5 has the results of the hyperparameter tuning with the grid and genetic algorithms. We found that the best configuration was founded by the genetic algorithm C, considering the values of the MSE, accuracy, precision and F1 metrics. Genetic algorithm B is also better than the hyperparameter grid result. On the other hand, the performance of genetic algorithm A is similar to that of the grid, it has worse MSE metric, but better accuracy, precision and F1 score.



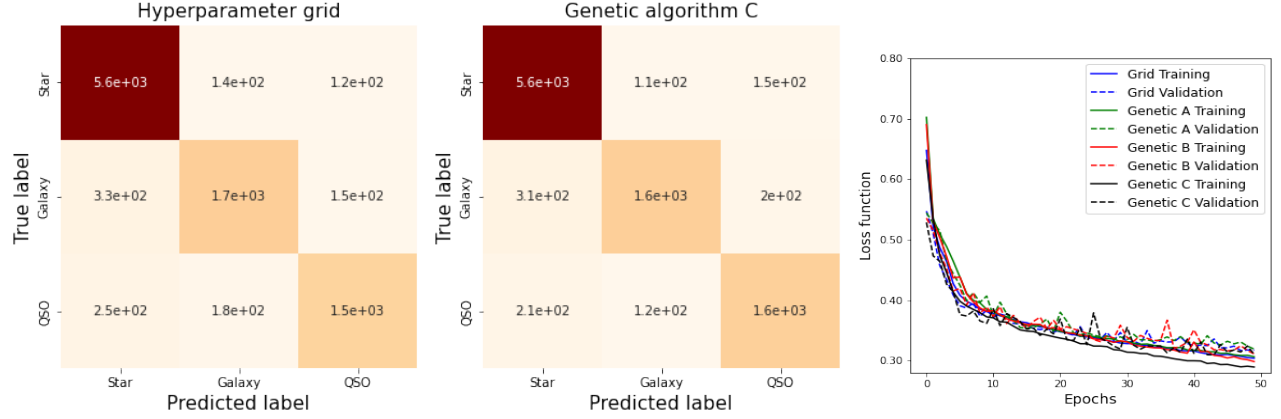


Figure 4: *Left and center*: Confusion matrix and loss functions to the SDSS-DR17 stellar objects classification performed with a grid of hyperparameters and the best genetic algorithm from Table 4. *Right*: Behavior of the loss functions.

During training, in two of three cases, the genetic algorithms have a lower value in the loss function and, from the loss functions plots in the Figure 4. On the other hand, using the test set, a slight advantage of the genetic algorithms over the hyperparameter network can be noted in all the classification metrics.

## 5 Discussion

Throughout the three cases in this work, we conclude that the use of the genetic algorithm is an interesting strategy to find a correct neural network architecture in a cosmological context. In the case where genetic algorithms are less advantageous, it is in the classification problem of Section 4.3; because the accuracy of the output of a neural network in a classification is not so crucial, as several values close to a class are rounded to that same class. However, in the example of the Section 4.1, where numerical accuracy is more relevant than in the other two cases, the advantage of the genetic algorithm over the hyperparameter grid is noticeable. It is noteworthy that when genetic algorithms use higher mutation probability values, fewer neural network architectures are evaluated, more variance is induced in individuals between subsequent generations and this may allow a good combination of hyperparameters to be found more quickly. We can notice the effect of the mutation probability in the number of evaluations, when it is small more neural network evaluations are needed.

We have observed that the hyperparameter grid evaluates each combination just once, regardless of its proximity to the optimal value. In contrast, genetic algorithms evaluate more times the configurations that are within a region of the search space where the optimum is likely to be, they may even evaluate the same point more than once; for example the mutation or crossover of two different individuals could generate the same new point. This behavior makes the solution found by the genetic algorithms more reliable because the individuals of the last population have been tested better than the configurations evaluated within the hyperparameter grid framework.

We tested the genetic algorithms with a relatively small values of population size and generations and, in the analyzed cases, their performance was very competitive with the traditional grid, and even better. In this paper we have tested hyperparameter tuning with genetic algorithms only with feedforward neural networks and in three very specific examples; nevertheless, this same methodology can be used in other cosmological applications and with any other type of neural network architecture. Moreover, in cosmological scenarios where more numerical precision, more complex architectures or larger search space size (i.e., more number of hyperparameters) are required, genetic algorithms are still expected to perform very well.

Another remark is that the process of running a genetic algorithm to find the hyperparameters of a neural network can be slow, similar to the hyperparameter grid method. However, in the current times of precision cosmology we believe it is a necessary cost to obtain better neural network models for the observational evidence.

## Acknowledgements

JAV acknowledges the support provided by FOSEC SEP-CONACYT Investigación Básica A1-S-21925, FORDECYT-PRONACES-CONACYT 304001 and UNAM-DGAPA-PAPIIT IA104221. IGV thanks the CONACYT postdoctoral fellowship, the support of the ICF-UNAM and Ricardo Medel Esquivel for the discussions on genetic algorithms.

## References

- [1] Ian Goodfellow, Yoshua Bengio, Aaron Courville, and Yoshua Bengio. *Deep learning*, volume 1. MIT press Cambridge, 2016.
- [2] Celia Escamilla-Rivera, Maryi A Carvajal Quintero, and Salvatore Capozziello. A deep learning approach to cosmological dark energy models. *Journal of Cosmology and Astroparticle Physics*, 2020(03):008. [arXiv:2109.00636], 2020.
- [3] Guo-Jian Wang, Xiao-Jiao Ma, Si-Yao Li, and Jun-Qing Xia. Reconstructing functions and estimating parameters with artificial neural networks: A test with a hubble parameter and sne ia. *Astrophysical Journal Supplement Series*, 246(1):13. [arXiv:1910.03636], 2020.
- [4] Isidro Gomez Vargas, Ricardo Medel Esquivel, Ricardo Garcia, and J Alberto Vazquez. Neural network reconstructions for the hubble parameter, growth rate and distance modulus. *SSRN Electronic Journal*, page [arXiv:2104.00595], 2022.
- [5] Hai-Nan Lin, Xin Li, and Li Tang. Non-parametric reconstruction of dark energy and cosmic expansion from the pantheon compilation of type ia supernovae. *Chinese Phys. C*, 43(7):075101. [arXiv:1905.11593], 2019.
- [6] Konstantinos Dialektopoulos, Jackson Levi Said, Jurgen Mifsud, Joseph Sultana, and Kristian Zarb Adami. Neural network reconstruction of late-time cosmology and null tests. *Journal of Cosmology and Astroparticle Physics*, 2022(02):023. [arXiv:2111.11462], 2022.
- [7] Philip Graff, Farhan Feroz, Michael P Hobson, and Anthony Lasenby. Bambi: blind accelerated multimodal bayesian inference. *Monthly Notices of the Royal Astronomical Society*, 421(1):169–180. [arXiv:1110.2997], 2012.
- [8] Hector J Hortua, Riccardo Volpi, Dimitri Marinelli, and Luigi Malago. Accelerating mcmc algorithms through bayesian deep networks. *arXiv preprint arXiv:2011.14276*, 2020.
- [9] Alessio Spurio Mancini, Davide Piras, Justin Alsing, Benjamin Joachimi, and Michael P Hobson. Cosmopower: emulating cosmological power spectra for accelerated bayesian inference from next-generation surveys. *Monthly Notices of the Royal Astronomical Society*, 511(2):1771–1788. [arXiv:2106.03846], 2022.
- [10] Isidro Gómez-Vargas, Ricardo Medel Esquivel, Ricardo García-Salcedo, and J Alberto Vázquez. Neural network within a bayesian inference framework. In *Journal of Physics: Conference Series*, volume 1723, page 012022. IOP Publishing, 2021.
- [11] Juan de Dios Rojas Olvera, Isidro Gómez-Vargas, and Jose Alberto Vázquez. Observational cosmology with artificial neural networks. *Universe*, 8(2):120. [arXiv:2112.12645], 2022.
- [12] Nathanaël Perraudin, Michaël Defferrard, Tomasz Kacprzak, and Raphael Sgier. DeepSphere: Efficient spherical convolutional neural network with healpix sampling for cosmological applications. *Astronomy and Computing*, 27:130–146, 2019.
- [13] Emille EO Ishida. Machine learning and the future of supernova cosmology. *Nature Astronomy*, 3(8):680–682, 2019.
- [14] A Moller and T De Boissiere. Supernnova: an open-source framework for bayesian neural network-based supernova classification. *Monthly Notices of the Royal Astronomical Society*, 491(3):4277–4293, 2020.
- [15] Stuart Geman, Elie Bienenstock, and René Doursat. Neural networks and the bias/variance dilemma. *Neural Comp.*, 4(1):1–58, 1992.
- [16] Hugo Larochelle, Dumitru Erhan, Aaron Courville, James Bergstra, and Yoshua Bengio. An empirical evaluation of deep architectures on problems with many factors of variation. *Proc. 24th Int. Conf. Machine Learning*, pages 473–480, 2007.
- [17] Frank Hutter, Holger H Hoos, Kevin Leyton-Brown, and Thomas Stützle. Paramils: an automatic algorithm configuration framework. *J. Artif. Intell. Res.*, 36:267–306, 2009.
- [18] Rémi Bardenet, Mátyás Brendel, Balázs Kégl, and Michèle Sebag. Collaborative hyperparameter tuning. *Proc. 30th Int. Conf. Machine Learning*, 28(2):199–207, 2013.

- [19] Xiang Zhang, Xiaocong Chen, Lina Yao, Chang Ge, and Manqing Dong. Deep neural network hyperparameter optimization with orthogonal array tuning. *Int. Conf. Neural Inf. Processing*, pages 287–295, 2019.
- [20] Piotr Wasiewicz and Krzysztof Hryniewicz. Optimization of quasar parametrization using genetic algorithms. 11176:944–952, 2019.
- [21] Rubén Arjona, Alessandro Melchiorri, and Savvas Nesseris. Testing the  $\Lambda$ cdm paradigm with growth rate data and machine learning. *Journal of Cosmology and Astroparticle Physics*, 2022(05):047. [arXiv:2107.04343], 2022.
- [22] Rubén Arjona. Machine learning meets the redshift evolution of the cmb temperature. *Journal of Cosmology and Astroparticle Physics*, 2020(08):009, 2020.
- [23] Alex Cole, Andreas Schachner, and Gary Shiu. Searching the landscape of flux vacua with genetic algorithms. *Journal of High Energy Physics*, 2019(11):1–39, 2019.
- [24] Lothar Terfloth and Johann Gasteiger. Neural networks and genetic algorithms in drug design. *Drug Discovery Today*, 6:102–108, 2001.
- [25] Amin Kabir Anaraki, Moosa Ayati, and Foad Kazemi. Magnetic resonance imaging-based brain tumor grades classification and grading via convolutional neural networks and genetic algorithms. *Biocybernetics and Biomedical Engineering*, 39(1):63–74, 2019.
- [26] Amin Kabir Anaraki, Moosa Ayati, and Foad Kazemi. Magnetic resonance imaging-based brain tumor grades classification and grading via convolutional neural networks and genetic algorithms. *biocybernetics and biomedical engineering*, 39(1):63–74, 2019.
- [27] Gloria Curilem, Jorge Vergara, Gustavo Fuentealba, Gonzalo Acuna, and Max Chacón. Classification of seismic signals at villarrica volcano (chile) using neural networks and genetic algorithms. *Journal of Volcanology and Geothermal Research*, 180(1):1–8, 2009.
- [28] Fabian Ruehle. Evolving neural networks with genetic algorithms to study the string landscape. *Journal of High Energy Physics*, 2017(8):1–20, 2017.
- [29] Kurt Hornik, Maxwell Stinchcombe, and Halbert White. Universal approximation of an unknown mapping and its derivatives using multilayer feedforward networks. *Neural networks*, 3(5):551–560, 1990.
- [30] Christopher M Bishop and Nasser M Nasrabadi. *Pattern recognition and machine learning*, volume 4. Springer, 2006.
- [31] Michael A Nielsen. *Neural networks and deep learning*, volume 25. Determination press San Francisco, CA, USA, 2015.
- [32] David E Rumelhart, Geoffrey E Hinton, and Ronald J Williams. Learning representations by back-propagating errors. *nature*, 323(6088):533–536, 1986.
- [33] Colin R Reeves. Genetic algorithms for the operations researcher. *INFORMS journal on computing*, 9(3):231–250, 1997.
- [34] Sourabh Katoch, Sumit Singh Chauhan, and Vijay Kumar. A review on genetic algorithm: past, present, and future. *Multimedia Tools and Applications*, 80(5):8091–8126, 2021.
- [35] E. Wirsansky. *Hands-On Genetic Algorithms with Python: Applying genetic algorithms to solve real-world deep learning and artificial intelligence problems*. Packt Publishing, 2020.
- [36] Kerry Gallagher and Malcolm Sambridge. Genetic algorithms: a powerful tool for large-scale nonlinear optimization problems. *Computers & Geosciences*, 20(7-8):1229–1236, 1994.
- [37] SN Sivanandam and SN Deepa. *Genetic algorithms*. Springer, 2008.
- [38] Geoffrey F Miller, Peter M Todd, and Shailesh U Hegde. Designing neural networks using genetic algorithms. In *ICGA*, volume 89, pages 379–384, 1989.
- [39] David J Montana, Lawrence Davis, et al. Training feedforward neural networks using genetic algorithms. In *IJCAI*, volume 89, pages 762–767, 1989.
- [40] Nurshazlyn Mohd Aszemi and PDD Dominic. Hyperparameter optimization in convolutional neural network using genetic algorithms. *International Journal of Advanced Computer Science and Applications*, 10(6), 2019.
- [41] Changxi Ma, Wei Hao, Fuquan Pan, and Wang Xiang. Road screening and distribution route multi-objective robust optimization for hazardous materials based on neural network and genetic algorithm. *PLoS one*, 13(6):e0198931, 2018.
- [42] Martín Abadi, Paul Barham, Jianmin Chen, Zhifeng Chen, Andy Davis, Jeffrey Dean, Matthieu Devin, Sanjay Ghemawat, Geoffrey Irving, Michael Isard, et al. {TensorFlow}: a system for {Large-Scale} machine learning. In *12th USENIX symposium on operating systems design and implementation (OSDI 16)*, pages 265–283, 2016.

- [43] François-Michel De Rainville, Félix-Antoine Fortin, Marc-André Gardner, Marc Parizeau, and Christian Gagné. Deap: A python framework for evolutionary algorithms. In *Proceedings of the 14th annual conference companion on Genetic and evolutionary computation*, pages 85–92, 2012.
- [44] Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.
- [45] Anton Valentinovich Ereemeev. A genetic algorithm with tournament selection as a local search method. *Journal of Applied and Industrial Mathematics*, 6(3):286–294, 2012.
- [46] Francisco Chicano, Andrew M Sutton, L Darrell Whitley, and Enrique Alba. Fitness probability distribution of bit-flip mutation. *Evolutionary Computation*, 23(2):217–248, 2015.
- [47] Félix-Antoine Fortin, François-Michel De Rainville, Marc-André Gardner, Marc Parizeau, and Christian Gagné. DEAP: Evolutionary algorithms made easy. *Journal of Machine Learning Research*, 13:2171–2175, 07 2012.
- [48] François-Michel De Rainville, Félix-Antoine Fortin, Marc-André Gardner, Marc Parizeau, and Christian Gagné. Deap: enabling nimbler evolutions. *ACM SIGEVOlution*, 6(2):17–26, 2014.
- [49] Varun Sahni and Alexei Starobinsky. Reconstructing dark energy. *Int. J. Mod. Phys. D*, 15(12):2105–2132, 2006.
- [50] Ranbir Sharma, Ankan Mukherjee, and HK Jassal. Reconstruction of late-time cosmology using principal component analysis. *preprint (arXiv:2004.01393)*, 2020.
- [51] Luis A Escamilla and J Alberto Vazquez. Model selection applied to non-parametric reconstructions of the dark energy. *arXiv preprint [arXiv:2111.10457]*, 2021.
- [52] Francesca Gerardi, Matteo Martinelli, and Alessandra Silvestri. Reconstruction of the dark energy equation of state from latest data: the impact of theoretical priors. *Journal of Cosmology and Astroparticle Physics*, 2019(07):042, 2019.
- [53] Christopher KI Williams and Carl Edward Rasmussen. *Gaussian processes for machine learning*, volume 2. MIT press Cambridge, MA, 2006.
- [54] Ryan E. Keeley, Arman Shafieloo, Gong-Bo Zhao, Jose Alberto Vazquez, and Hanwool Koo. Reconstructing the universe: Testing the mutual consistency of the pantheon and SDSS/eBOSS BAO data sets with gaussian processes. *The Astronomical Journal*, 161(3):151. [arXiv:2010.03234], 2 2021.
- [55] Benjamin L’Huillier, Arman Shafieloo, David Polarski, and Alexei A Starobinsky. Defying the laws of gravity i: model-independent reconstruction of the universe expansion from growth data. *Monthly Notices of the Royal Astronomical Society*, 494(1):819–826, 2020.
- [56] Purba Mukherjee and Narayan Banerjee. Revisiting a non-parametric reconstruction of the deceleration parameter from combined background and the growth rate data. *Physics of the Dark Universe*, 36:100998, 2022.
- [57] Ariadna Montiel, Ruth Lazkoz, Irene Sendra, Celia Escamilla-Rivera, and Vincenzo Salzano. Nonparametric reconstruction of the cosmic expansion with local regression smoothing and simulation extrapolation. *Physical Review D*, 89(4):043007, 2014.
- [58] J.Alberto Vazquez, M. Bridges, M.P. Hobson, and A.N. Lasenby. Reconstruction of the Dark Energy equation of state. *Journal of Cosmology and Astroparticle Physics*, 09:020. [arXiv:1205.0847], 2012.
- [59] Sonke Hee, JA Vázquez, WJ Handley, MP Hobson, and AN Lasenby. Constraining the dark energy equation of state using bayes theorem and the kullback–leibler divergence. *Monthly Notices of the Royal Astronomical Society*, 466(1):369–377. [arXiv:1607.00270], 2017.
- [60] Savvas Nesseris and Juan Garcia-Bellido. A new perspective on dark energy modeling via genetic algorithms. *Journal of Cosmology and Astroparticle Physics*, 2012(11):033, 2012.
- [61] M et al Betoule, R Kessler, J Guy, J Mosher, D Hardin, R Biswas, P Astier, P El-Hage, M Konig, S Kuhlmann, et al. Improved cosmological constraints from a joint analysis of the sdss-ii and snls supernova samples. *Astronomy & Astrophysics*, 568:A22, 2014.
- [62] I.E. Lagaris, A. Likas, and D.I. Fotiadis. Artificial neural networks for solving ordinary and partial differential equations. *IEEE Transactions on Neural Networks*, 9(5):987–1000, 1998.
- [63] Tamirat Temesgen Dufera. Deep neural network for system of ordinary differential equations: Vectorized algorithm and simulation. *Machine Learning with Applications*, 5:100058, 2021.
- [64] Miguel A Aragon-Calvo. Classifying the large-scale structure of the universe with deep neural networks. *Monthly Notices of the Royal Astronomical Society*, 484(4):5771–5784. [arXiv:1804.00816], 2019.
- [65] AK Aniyani and Kshitij Thorat. Classifying radio galaxies with the convolutional neural network. *The Astrophysical Journal Supplement Series*, 230(2):20, 2017.

- [66] Ansh Mittal, Anu Soorya, Preeti Nagrah, and D Jude Hemanth. Data augmentation based morphological classification of galaxies using deep convolutional neural network. *Earth Science Informatics*, 13(3):601–617, 2020.
- [67] MC Storrie-Lombardi, O Lahav, L Sodre Jr, and LJ Storrie-Lombardi. Morphological classification of galaxies by artificial neural networks. *Monthly Notices of the Royal Astronomical Society*, 259(1):8–12., 1992.
- [68] Katherine Accetta, Conny Aerts, Víctor Silva Aguirre, Romina Ahumada, Nikhil Ajgaonkar, N Filiz Ak, Shadab Alam, Carlos Allende Prieto, Andrés Almeida, Friedrich Anders, et al. The seventeenth data release of the sloan digital sky surveys: Complete release of manga, mastar, and apogee-2 data. *The Astrophysical Journal Supplement Series*, 259(2):35, 2022.