DRAFT VERSION APRIL 28, 2022
Typeset using LaTeX **preprint** style in AASTeX63

Classifying X-ray Active Galactic Nuclei Using TensorFlow Decision Forests and Neural Networks

NIKKO J. CLERI[1, 2]

[1]*Department of Physics and Astronomy, Texas A&M University, College Station, TX, 77843-4242 USA*
[2]*George P. and Cynthia Woods Mitchell Institute for Fundamental Physics and Astronomy, Texas A&M University, College Station, TX, 77843-4242 USA*

ABSTRACT

Modern statistical learning has had an enormous impact on the progress of data-driven sciences. In this work, we discuss the usability of TensorFlow, an open source end-to-end machine learning platform, and apply TensorFlow methodologies to real data. We apply both decision forest and neural network models to classify the X-ray activity of 1574 objects from the Chandra Deep Field North and Chandra Deep Field South X-ray catalogs. We find that our neural network, random forest, and gradient-boosted decision forest algorithms all classify the X-ray AGN data with an accuracy of ≈90%. We also discuss the implications of machine learning and statistical computing in the future of astronomy with next-generation observatories like the *James Webb Space Telescope*, *Nancy Grace Roman Space Telescope*, and *Vera C. Rubin Observatory*.

1. INTRODUCTION

In the modern era, sciences of all disciplines are becoming increasingly data-driven. With experiments and observations producing such enormous amounts of data, and next-generation experiments to only produce even more, analysts of these data require advanced methods of processing.

The advent of statistical learning methods through often complex computational algorithms (often referred to as "machine learning" or "ML") has introduced a much-needed tool into the realm of data analysis. Several different mechanisms have arisen in recent years to apply machine learning techniques to scientific datasets.

In this work, we consider the use of the modern machine learning platform TensorFlow Abadi et al. (2015). TensorFlow offers capabilities widely useful in many areas of science and industry demanding efficient and accessible machine learning. We specifically consider use cases in astrophysics, a rapidly growing field in high need of a major revolution in large-scale data processing.

In astrophysics, data are being produced by current generation observatories at a rate greater than the growth of the storage supply. In the soon-to-be generation of telescopes including the *Vera C. Rubin Observatory* (*VRO*), *Nancy Grace Roman Space Telescope* (*NGRST*), and the recently-launched *James Webb Space Telescope* (*JWST*), the production of data will greatly exceed the on-board storage capacity. These observatories will require some means of computationally efficient data preprocessing to avoid losing valuable information.

Machine learning methods for data processing are rising in popularity in astronomy research. Artificial neural networks (ANNs) have been used to classify stellar spectra as far back as 1998 (Singh et al. 1998), determining photometric redshifts (Firth et al. 2003; Way & Klose 2012), and other applications. Extremely randomized trees (ETs) have been used to estimates physical parameters of stellar atmospheres (Zhang et al.

2019) and in redshift estimation of nearby galaxies and distant quasars (Reza & Haque 2020). Support-vector machines (SVMs) have been used to classify spectral subclasses (Bu et al. 2014).

One of the most common widely studied problems in astrophysics that has benefited greatly from the advent of machine learning is the classification of galaxy morphologies. This has been been studied over the last three decades using several different machine learning algorithms: feedforward neural networks, two-hidden-layer ANNs, random forests and decision trees, along with SVM, and K-means clustering and agglomerative hierarchical clustering (Storrie-Lombardi et al. 1992; Banerji et al. 2010; Gauci et al. 2010; Sreejith et al. 2018; Reza 2021).

In an example use case of TensorFlow, we use artificial neural networks along with both random and gradient-boosted decision forests to classify the activity of galaxies from the Chandra Deep Field Surveys (Xue et al. 2016; Luo et al. 2017). This is a simple problem compared to the incredibly deep and complex data sets which will come from the upcoming surveys by next generation observatories, but serves as a concise proof of concept for such work done with TensorFlow machine learning models in Python.

The remainder of this work is as follows. Section 2 discusses an overview of the usability and functions of TensorFlow in Python. Section 3 shows a use case of TensorFlow on a real astrophysical data, in a study to classify X-ray detected active galactic nuclei. Section 4 discusses the implications of TensorFlow in the evolving picture of data-driven sciences.

2. OVERVIEW OF AND INTRODUCTION TO TENSORFLOW

In this section, we discuss the methodology of TensorFlow and its fundamental usage. TensorFlow is an open source end-to-end platform designed for machine learning, developed notably for Python, Javascript, Java, and C++, developed by Google and initially released in 2015 (Abadi et al. 2015).

TensorFlow is widely accessible and optimizable for a range of computing hardware. TensorFlow can be configured for central processing units (CPUs), graphical processing units (GPUs), or custom Google-manufactured integrated circuits specifically designed to perform machine learning computations with TensorFlow. These tensor processing units (TPUs) are optimized to perform calculations on multidimensional arrays, or tensors, the fundamental building block of the TensorFlow paradigm.

The cornerstone of modern machine learning with TensorFlow and other platforms is tensor calculus, especially algorithms such as gradient descent (or steepest descent). TensorFlow employs automatic differentiation, which is essentially the repeated application of the chain rule to fundamental arithmetic operations and functions to compute derivatives of arbitrary order. TensorFlow computes gradients by storing the operations used in the "forward mode" of the automatic differentiation of a function or model and traverses the same operations in reverse order in the "reverse mode". The automatic differentiation to calculate a gradient is most often used to compute the error or loss of a model with respect to its weights.

TensorFlow machine learning has been applied to a wide variety of real-data scenarios across science and industry. Several examples of these applications include: video and audio processing and classification, lingustics via text/speech recognition and translation, time series data analysis in meteorology, and may others. The TensorFlow platform also works well with structured data analysis, much like the data we see in the natural sciences.

TensorFlow operates in Python using implementation from the algorithm library Keras (Chollet et al. 2015). TensorFlow/Keras offers several different algorithms for machine learning with a wide variety of data. In this work, we explore two primary modes of machine learning: neural networks (NNs) and decision forests.

3. TENSORFLOW USE CASE: CLASSIFYING X-RAY AGN ACTIVITY FROM THE CDFS/CDFN CATALOGS

In this section, we apply the methodology of TensorFlow to a use case with real data. We apply both neural network and decision forest models to these data in a binary classification problem. We study the classification of the activity of X-ray detected objects in the Chandra Deep Field North (CDFN) and Chandra Deep Field South (CDFS) catalogs (Xue et al. 2016; Luo et al. 2017).

3.1. *Physical Description of Active Galactic Nuclei*

Here we offer the physical background of the classification problem studied in this work. In extragalactic astronomy, it is widely accepted that most, if not all, massive galaxies host a supermassive black hole (SMBH) in their centers (Kormendy & Richstone 1995; Kormendy & Ho 2013). In many cases, the SMBH will accrete surrounding matter in the form of dust, gas, and stars. Should the accretion rate be high enough, the dynamical frictions and other forces around the "central engine" produce an enormous amount of energy. This high-energy accretion releases radiation with an identifiable signature, which can then be studied using spectroscopy and photometry (Seyfert 1943; Baldwin et al. 1981; Veilleux & Osterbrock 1987; Kewley et al. 2006; Trump et al. 2011, 2015; Backhaus et al. 2022; Xue et al. 2016; Luo et al. 2017). These exotic high-energy objects are referred to "active" galactic nuclei (AGN) (Seyfert 1943; Kormendy & Richstone 1995; Kormendy & Ho 2013).

3.2. *Data: Chandra X-ray Catalogs*

The CDFN+CDFS catalogs have 1691 objects with X-ray detections. The sample is limited by minimum detectable fluxes in the 0.5-2 keV (soft) band and 2-8 keV (hard) bands are $\approx 1.3 \times 10^{-16}$ erg cm^2 s^s and $\approx 6.5 \times 10^{-16}$ erg cm^2 s^s, respectively (Hornschemeier et al. 2001). Each object is classified in to one of three classes: active galactic nuclei (AGN), galaxies, or stars. An object is classified as an AGN if it satisfies at least one of the following criteria:

- $L_{0.5\text{–}7\ \mathrm{keV}} \geq 3 \times 10^{42}$ erg s^{-1} (Luminous AGN)
- Effective photon index $\Gamma \leq 1.0$ (Obscured AGN)
- X-ray-to-optical flux ratio of $\log(f_X/f_R) > -1$ where $f_x = f_{0.5\text{–}7\ \mathrm{keV}}, f_{0.5\text{–}2\ \mathrm{keV}}$ or $f_{2\text{–}7\ \mathrm{keV}}$
- Excess X-ray emission over expectation from pure star formation $L_{0.5\text{–}7\ \mathrm{keV}} \geq 3 \times (8.9 \times 10^{17}\ L_R)$

Objects labeled as "Galaxies" in the X-ray catalogs are are those which are confirmed to be galaxies but do not meet these criteria. "Stars" are classified by having 0 or unphysical redshifts.

We reduce this classification problem to binarity by defining a classification of AGN by these criteria to be a "success" (value 1) and all other objects ("galaxies" or "stars") to be "failures" (value 0).

The effective photon index Γ is defined by

$$N = AE^{-\Gamma} \tag{1}$$

where N is the is the number of photons per square centimeter per keV, A is a normalization constant, and E is the hard band energy (Hornschemeier et al. 2001).

We choose to ignore some other predictors offered in the Xue et al. (2016) and Luo et al. (2017) catalogs which have sparse detections. We perform such cuts because certain TensorFlow neural network algorithms are not compatible with nonexistent predictors for a subset of the data.

Table 1. Descriptions of Predictors from CDFN/CDFS Catalogs

Predictor	Description
`log_Lx`	$\log(L_{\mathrm{X-ray}})$ in Full Band (erg/s)
`zadopt`	Adopted Redshift (spectroscopic where available, photometric otherwise)
`PInd`	Effective Photon Index Γ (see Equation 1)
`SExp`	Soft Band Exposure Time (in seconds)
`HExp`	Hard Band Exposure Time (in seconds)
`FExp`	Full Band Exposure Time (in seconds)
`SFlux`	Soft Band Flux (in erg/s/cm^2)
`HFlux`	Hard Band Flux (in erg/s/cm^2)
`FFlux`	Full Band Flux (in erg/s/cm^2)
`BRat`	Hard/Soft Band Flux Ratio

We cut objects from the sample with unmeasured predictors, which are not compatible with the models. This leaves us with a final sample of 1574 objects. The Chandra X-ray catalogs classify 78% of these 1574 objects as AGN (1224).

Figure 1 shows the correlations of the ten predictors used in the following analysis. The diagonal elements show the kernel density estimations of the distribution of the respective parameter. We note that there are some (by definition) correlated parameters. The hard, soft, and full band fluxes are correlated with each other along with the hard, soft, and full band exposure times. There is also a correlation between the hard/soft band flux ratios and the effective photon indices Γ, which is an artefact of the underlying physics of the photon index.

3.3. *Description of Neural Network Models*

Here we describe the applications of the neural network model used in this classification problem. In general, we consider the inner workings of an artificial neural network (ANN) to be a black box. The input data is normalized and fed through the hidden layers of the neural network to achieve an output prediction.

Figure 2 shows the TensorFlow schematic of the neural network used. We normalize our data to be usable by the TensorFlow neural networks using the `tf.keras.layers.Normalization` preprocessing layer, which shifts and scales inputs into a distribution centered about 0 with a standard deviation of 1. The mean and standard deviation are calculated at runtime using the `adapt` method. The normalization improves the accuracy and computational efficiency of the neural network training (Sola & Sevilla 1997).

The important tunable hyperparameters of the TensorFlow/Keras neural network models include the activation function, the depth (number of layers), optimization function, loss function, and dimensionality of output layer, among others. This suite of tunable parameters allows for great amounts of customization and specialization for a wide variety of problems.

The `activation` hyperparameter in our neural network is given by a sigmoid function:

$$f(x) = \frac{1}{(1+e^{-x})} \tag{2}$$

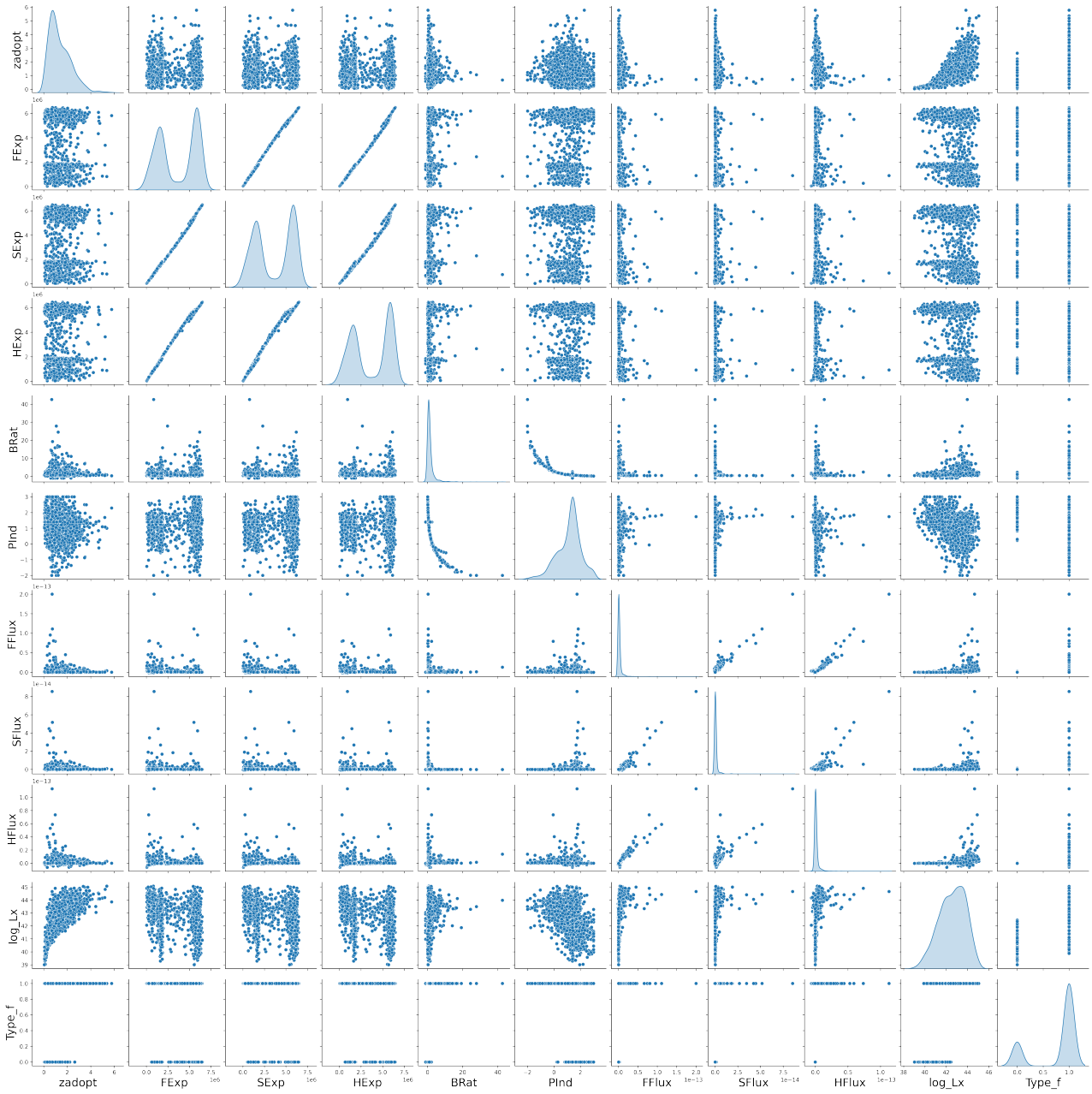

Figure 1. Pair plot of predictors used in this example analysis. The diagonal elements show the kernel density estimations of the distribution of the respective parameter. We see that there are some (by definition) covariant parameters, including the exposure times and fluxes in the full band. We also see an exponential relationship between the hard/soft band ratio and the effective photon index Γ, which is an artefact of the underlying physics of the AGN obscurity parameterized by the effective photon index.

The loss function used is that of binary cross-entropy, given by

$$\text{Loss} = -\frac{1}{N}\sum_{i=1}^{N} y_i * \log(p(y_i)) + (1-y_i)\log(1-p(y_i)) \tag{3}$$

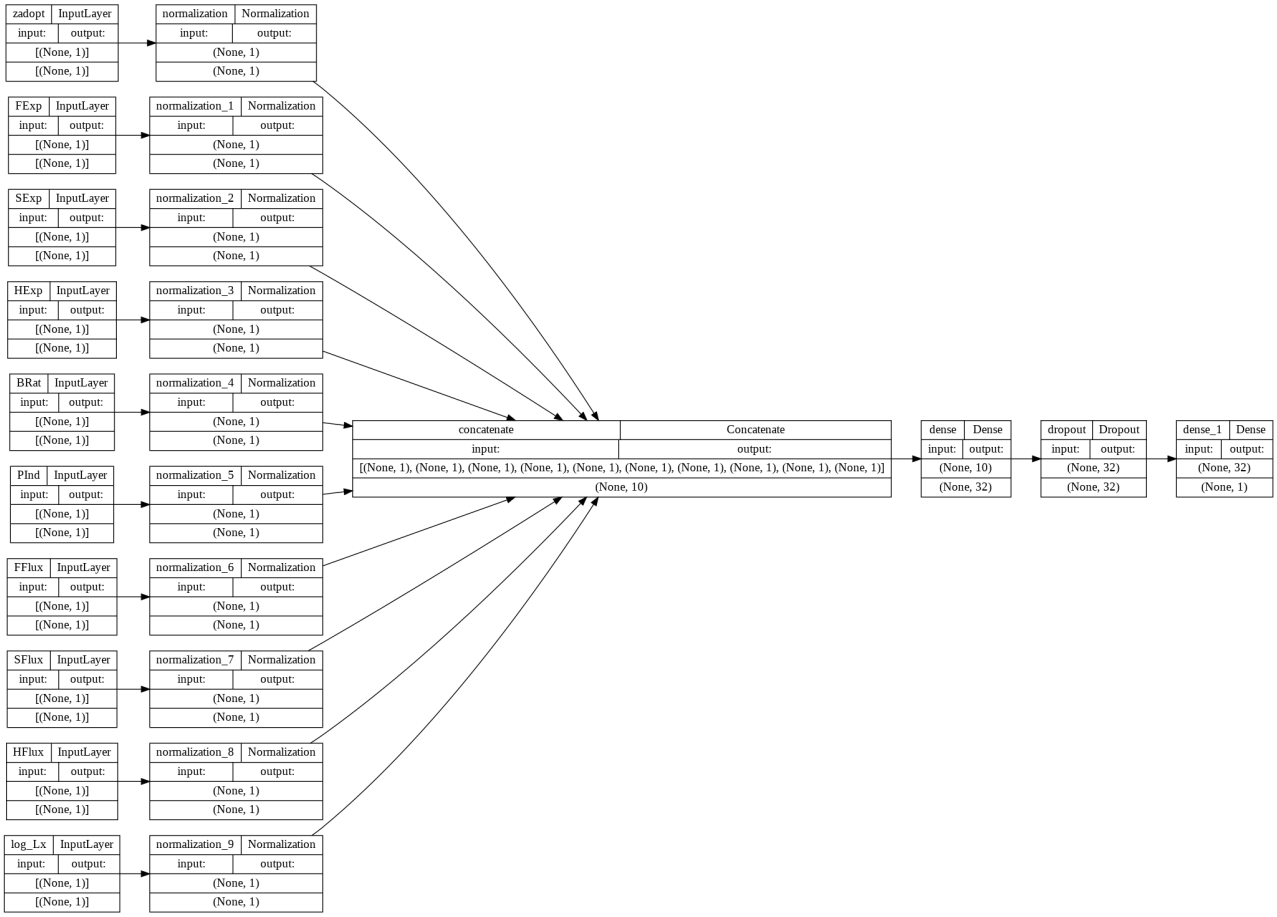

Figure 2. TensorFlow schematic of the neural network used for our AGN classification problem. The 10 input predictors are normalized in the process described in 3.3. The normalized inputs are then passed through the hidden layers of the neural network which result in the output layer.

We also implement a dropout rate of 0.5, where the dropout layer randomly drops units at a rate of 0.5 at each step during training to prevent overfitting. The remaining inputs at each step in training are scaled a factor of 2 to maintain the sum of all inputs at each step.

We consider the hidden layers of the neural network to be a black box with the given configuration and only draw conclusions based on the predictions from the output.

The hyperparameters used in our neural network analysis are shown in Table 1. These are largely the default parameters for a TensorFlow/Keras neural network built for binary classification.

3.3.1. *Neural Network Results*

We divide our sample into training (80%) and testing (20%) subsamples. Our model achieves an accuracy of greater than 90% in the training data set and and accuracy of 87% in the validation set. Figure 3 shows the accuracy and loss of the validation set as a function of the number of epochs. We note that our model converged very quickly (less than 10 epochs) to above 85% accuracy.

As described earlier, the neural network behaves effectively as a black box. We cannot use the results of the neural network in Figure 3 to draw any conclusions other than the predictive efficacy of the model. We conclude that this neural network is a fast and efficient means of prediction for this problem, but offers

Table 1. Hyperparameters used in Neural Network Analysis

Parameter Description	Used Hyperparameter
Activation Function	Sigmoid (see Equation 2)
Optimization Function	Adam Algorithm (Kingma & Ba 2014)
Loss Function	Binary Cross-Entropy (see Equation 3)
Output Dimensionality	32
Dropout Rate	0.5
Number of Epochs	100

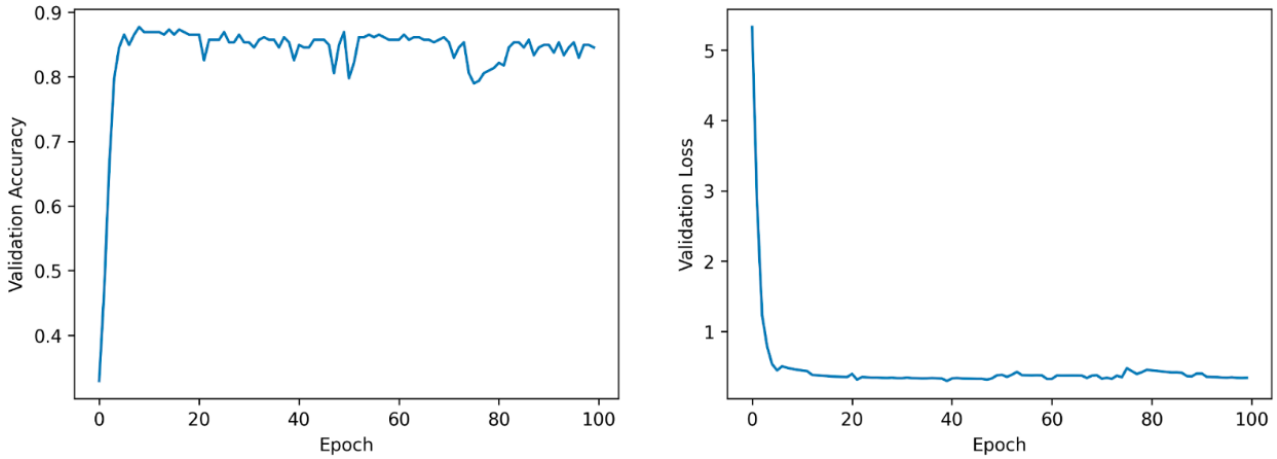

Figure 3. Results of neural network analysis. *Left*: Accuracy of the model for the validation set as a function of the number of epochs. *Right*: Loss calculated by binary cross-entropy as a function of epoch. We note that the model converges to greater than 85% accuracy in the validation set in less than 10 epochs.

nothing in terms of interpretation. In the following section, we present the results of the decision forest analysis, which offers similar accuracy with some insight into the inner workings.

3.4. *Description of Decision Forest Models*

Here we describe the applications of the decision forest models used in this classification problem. Decision forests are comprised of an ensemble of unique decision trees. Each decision tree operates by making a decision at each node to purify the output at each step. This purification process can be parameterized in terms of a number of metrics, commonly entropy loss per decision or a parameter known as the Gini coefficient:

$$G = \sum_{k=1}^{K} \hat{p}_{mk}(1 - \hat{p}_{mk}) \tag{4}$$

In this work, we consider the loss of entropy, given by

$$D = -\sum_{k=1}^{K} \hat{p}_{mk} \log \hat{p}_{mk} \tag{5}$$

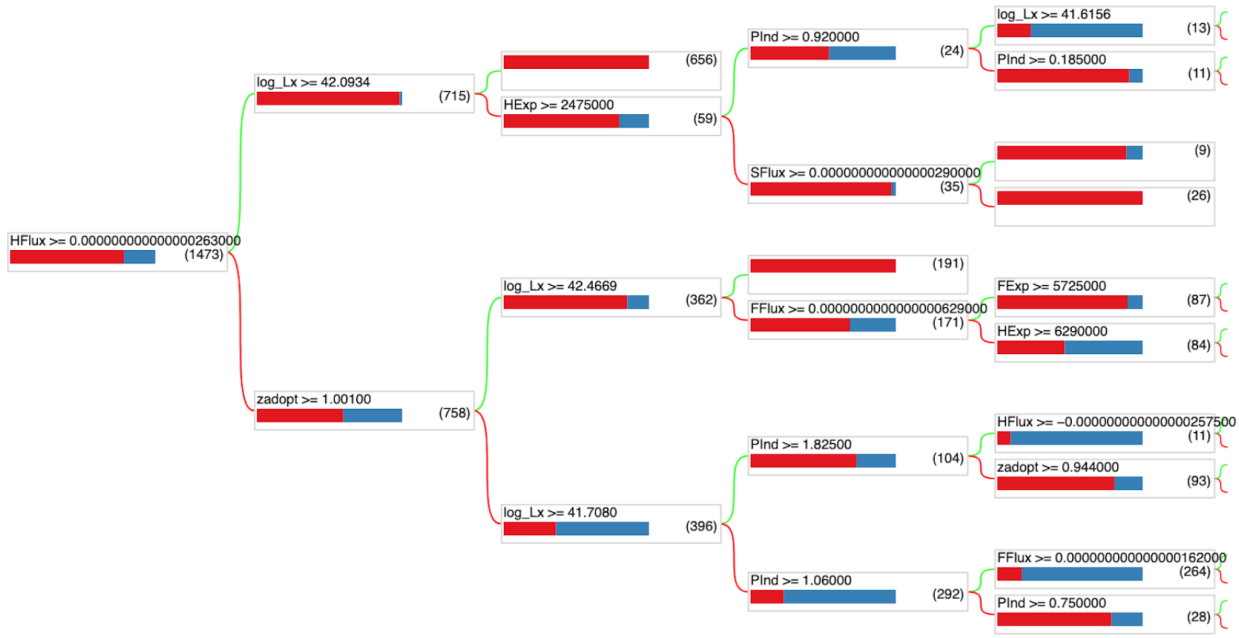

Figure 4. TensorFlow schematic of an example decision tree in our random forest model. The decisions are made to purify the output using an entropy loss function in a greedy nature as described in Section 3.4. The red portion of the bar in each leaf represents the amount of the sample at that step classified as AGN (1) where the blue represents the amount classified as a non-AGN (0).

where $\hat{p}_{mk}$ is defined the same as in the Gini index equation.

Any parameterization of such a purity index attempts to categorize the sample at each decision in a greedy nature. The decision trees continue purifying the sample by minimizing the entropy at each node until a final decision is made for that tree. The average of the decisions results in the final decision of the forest. The use of a decision forest over individual trees allows for greater accuracy in testing data by avoiding overfitting (James et al. 2013).

The decision forest models in TensorFlow/Keras used in this work are divided into two major categories: Random Forests (RFs) and Gradient-Boosted Decision Forests (GBDFs).

The random forest model applies a bootstrapping technique to form a forest of n decision trees. The data used to train each tree are sample from the training data with replacement, allowing each tree to be trained on a unique subset. The nodes of each tree are made using a random subset of predictors, which ensures the uniqueness of the individual trees, thus, the forest is randomized. This helps in reducing overfitting compared to individual deterministic trees.

The gradient-boosted decision forest model operates in a similar manner, except each tree is created from the fit of the residuals of the previous tree. This essentially "zooms in" on the problematic nodes of the tree, where the relevant loss function is minimized least effectively (James et al. 2013). This model is designed to interpret data with large amounts of predictors and data with missing entries (Friedman 2000).

Figure 4 shows the TensorFlow schematic of an example decision tree from one of our decision forest models used.

We employ four different decision forest models in our analysis. The details of each model are shown in Table 2. Each model has a forest size of 300 trees.

Table 2. Decision Forest Model Descriptions

Model	Parameters	K-fold Validation Accuracy
Random Forest	Default Parameters	0.900
Gradient-Boosted	Default Parameters	0.904
Gradient-Boosted	Best First Global growing strategy	0.910
Gradient-Boosted	Sparse Oblique axis splitting (Tomita et al. 2020)	0.920

3.4.1. *Decision Forest Results*

We present the accuracy and loss as a function of forest size number of trees in Figure 5. Both the RF and GBDF models produce a similar accuracy, around 90%. We see that the GBDF model, when allowed to run for a maximum of 300 trees, overfits the data and truncates itself. To prevent this, we show a GBDF model with a maximum of 20 trees.

The relative importance of predictors from the decision forest models can be calculated by the relative entropy loss at each decision using the respective predictor. We show the results of this calculation in Figure 6. We note that, by a large margin, the most important predictor is the X-ray luminosity, followed by the effective photon index Γ. This is to be expected given the definitions of AGN criteria from the Xue et al. (2016) and Luo et al. (2017) catalogs in Section 3.2. We consider this a success in the models ability to discern the known important predictors of the data.

3.4.2. *K-Fold Validation of Decision Forest Models*

We perform a K-fold validation to confirm the efficacy of our models. We choose a 10-fold cross-validation with the `sklearn.model_selection.KFold` method (Pedregosa et al. 2011). Each model was trained 10 times with 9 of the 10 groups. The remaining group was used to calculate the validation accuracy. Accuracy here is calculated by number correct divided by number in validation set. Table 2 shows the k-fold validation accuracy of each of our four decision forest models.

The RF and GBDF models all achieved similar accuracy in the k-fold validation. We conclude that there is no meaningful return in using a more complicated, more computationally expensive model for these data. Given this, we choose the default GBDF model as our most reasonable decision forest model to use for this study.

4. CONCLUSIONS

In this work, we have discussed an overview and use case of TensorFlow, an end-to-end machine learning platform which is widely used in the physical sciences. We discuss the application and usage of TensorFlow and its Python algorithm library Keras, and apply neural network and decision forest models to an application in astronomy with binary AGN classification using data from the Chandra Deep Field X-ray catalogs (Xue et al. 2016; Luo et al. 2017). We present our primary conclusions here:

- Our neural network analysis achieves a validation accuracy of over 90%. We consider the ANN paradigm to be an effective, computationally efficient model for prediction for this problem. However, we cannot draw any conclusions of inference or interpretation, and as such consider the inner workings of the neural network to be a black box.

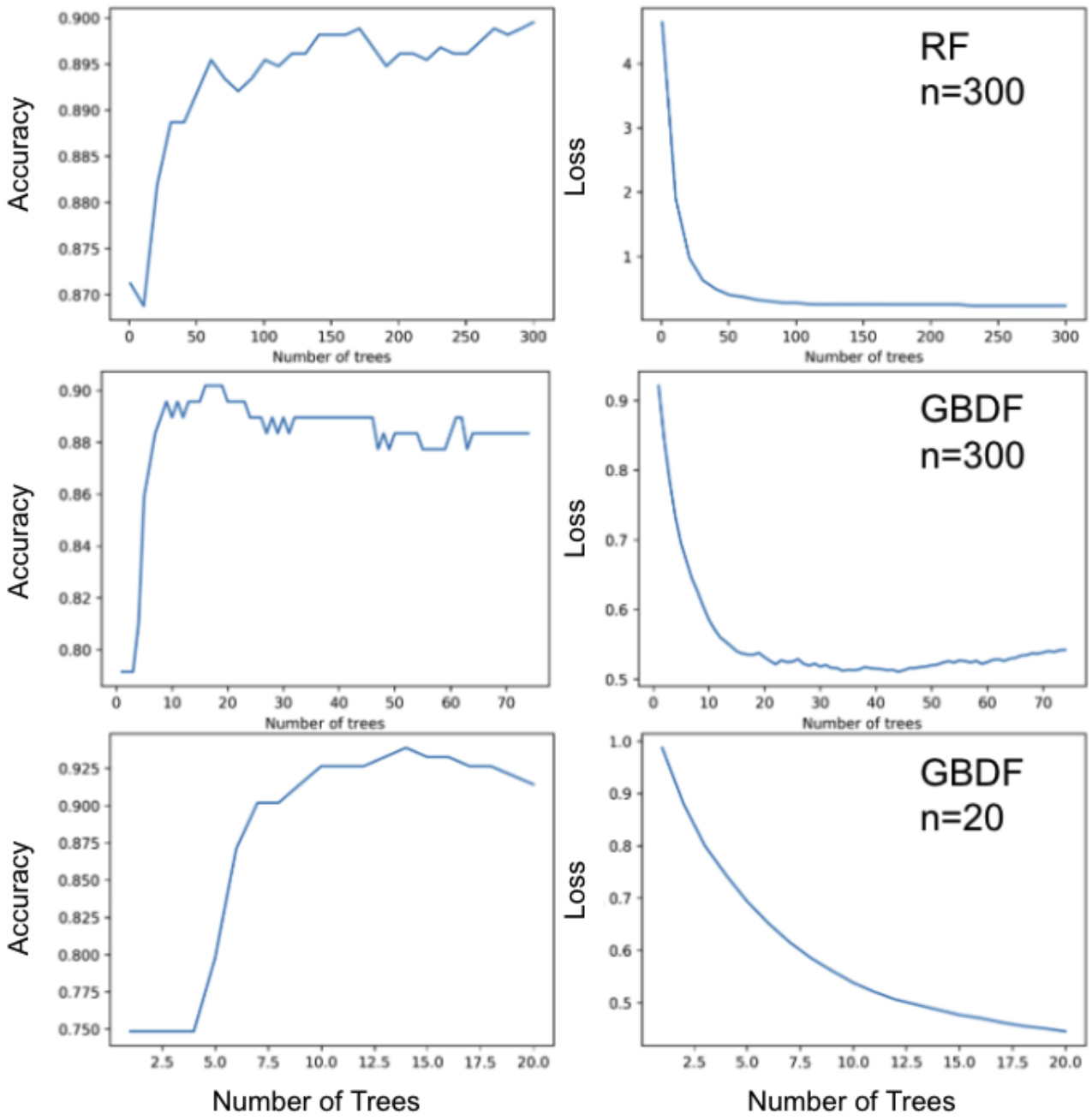

Figure 5. Accuracy and loss as a function of number of trees for our random forest model (*top*) and default gradient boosted decision forest model (*center*), each with a maximum of 300 trees. The GBDF model is designed to truncate if it overfits the data, as shown in the center panels with the increasing loss function after ∼40 trees. We also show a GBDF model where we only allow a maximum of 20 trees to prevent overfitting.

- We test several decision forest models, of both the random and gradient-boosted varieties, and find that they all achieve similar $\sim 90\%$ accuracy with k-fold validation. We conclude that the simplest gradient-boosted decision forest is the optimal DF model for this work, given that it achieves similar accuracy but is less computationally expensive than the other DF models.

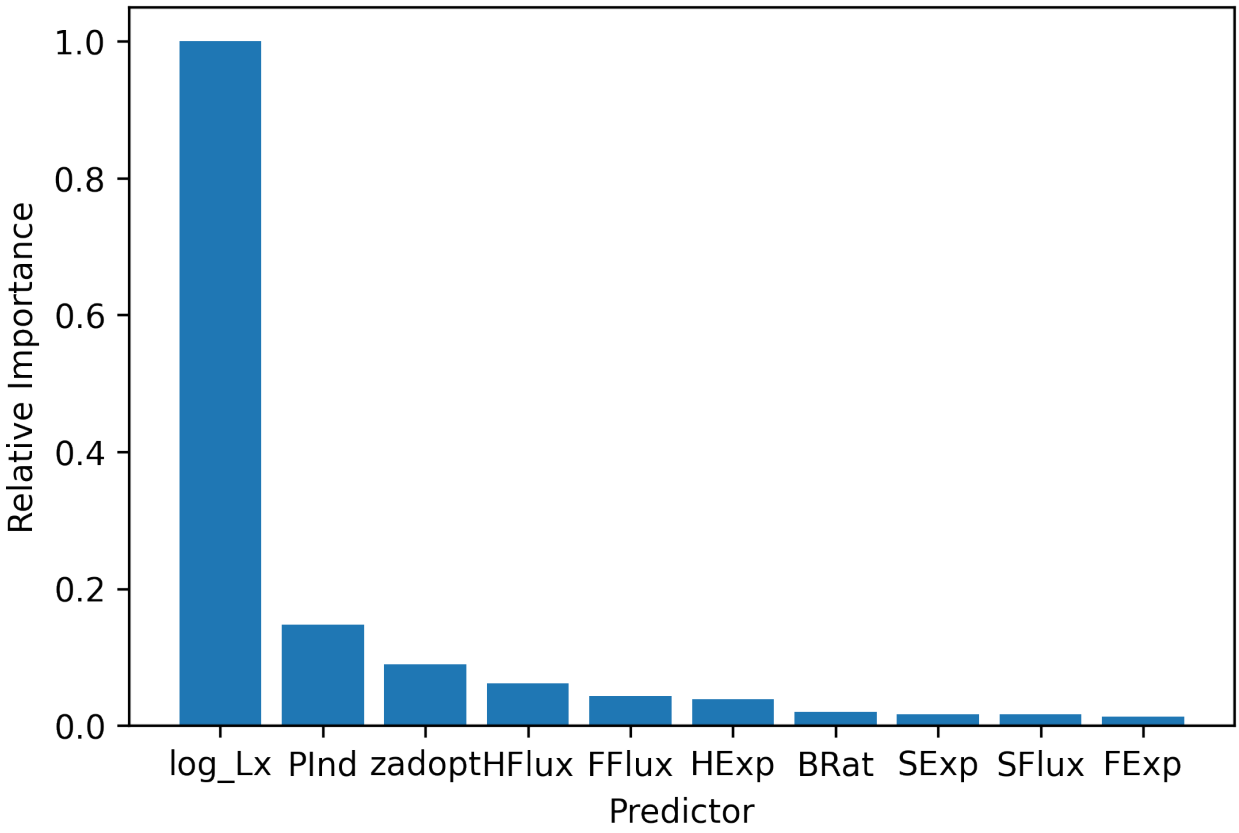

Figure 6. The relative importance of each of our 10 predictors in the 20 tree gradient boosted decision forest model. We quantify relative importance by the relative entropy loss by each decision using the respective predictor. The X-ray luminosity (`log_Lx`) is the most importance predictor followed by the effective photon index Γ (`PInd`). As discussed in section 3.2, the Xue et al. (2016) and Luo et al. (2017) catalogs define an AGN in terms of their effective photon indices and X-ray luminosities, so this result is a confirmation that the decision forest models are recognizing the importance of the most meaningful predictors.

- The most important predictors in our decision forest analyses are the X-ray luminosity and effective photon index. We consider this a success of the models to recover the predictors used in the definitions of the AGN classification from the Chandra catalogs.

This work is a relatively simple case which shows how useful these machine learning techniques can be for the future of large astronomical datasets. Each method used (neural network, random forest, gradient-boosted decision forest) offer high accuracy in the classification of these X-ray AGN. In the future, the implementation of computationally efficient machine learning algorithms will be a necessity in astronomy, with next-generation observatories like the *Vera C. Rubin Observatory*, *James Webb Space Telescope*, and *Nancy Grace Roman Space Telescope* producing orders of magnitude more data than current surveys.

REFERENCES

Abadi, M., Agarwal, A., Barham, P., et al. 2015, TensorFlow: Large-Scale Machine Learning on Heterogeneous Systems, , , software available from tensorflow.org. https://www.tensorflow.org/

Backhaus, B. E., Trump, J. R., Cleri, N. J., et al. 2022, ApJ, 926, 161

Baldwin, J. A., Phillips, M. M., & Terlevich, R. 1981, PASP, 93, 5
Banerji, M., Lahav, O., Lintott, C. J., et al. 2010, MNRAS, 406, 342
Bu, Y., Chen, F., & Pan, J. 2014, NewA, 28, 35
Chollet, F., et al. 2015, Keras, https://keras.io, ,
Firth, A. E., Lahav, O., & Somerville, R. S. 2003, Monthly Notices of the Royal Astronomical Society, 339, 1195. https://doi.org/10.1046%2Fj.1365-8711.2003.06271.x
Friedman, J. 2000, The Annals of Statistics, 29, doi:10.1214/aos/1013203451
Gauci, A., Zarb Adami, K., & Abela, J. 2010, arXiv e-prints, arXiv:1005.0390
Hornschemeier, A. E., Brandt, W. N., Garmire, G. P., et al. 2001, ApJ, 554, 742
James, G., Witten, D., Hastie, T., & Tibshirani, R. 2013, An Introduction to Statistical Learning: with Applications in R, Springer Texts in Statistics (Springer New York), doi:10.1007/978-1-4614-7138-7. http://doi.org/10.1007/978-1-4614-7138-7
Kewley, L. J., Groves, B., Kauffmann, G., & Heckman, T. 2006, MNRAS, 372, 961
Kingma, D. P., & Ba, J. 2014, Adam: A Method for Stochastic Optimization, arXiv, doi:10.48550/ARXIV.1412.6980. https://arxiv.org/abs/1412.6980
Kormendy, J., & Ho, L. C. 2013, ARA&A, 51, 511
Kormendy, J., & Richstone, D. 1995, ARA&A, 33, 581
Luo, B., Brandt, W. N., Xue, Y. Q., et al. 2017, The Astrophysical Journal Supplement Series, 228, 2
Pedregosa, F., Varoquaux, G., Gramfort, A., et al. 2011, Journal of Machine Learning Research, 12, 2825
Reza, M. 2021, Astronomy and Computing, 37, 100492
Reza, M., & Haque, M. A. 2020, Ap&SS, 365, 50
Seyfert, C. K. 1943, ApJ, 97, 28
Singh, H. P., Gulati, R. K., & Gupta, R. 1998, MNRAS, 295, 312
Sola, J., & Sevilla, J. 1997, IEEE Transactions on Nuclear Science, 44, 1464
Sreejith, S., Pereverzyev, Sergiy, J., Kelvin, L. S., et al. 2018, MNRAS, 474, 5232
Storrie-Lombardi, M. C., Lahav, O., Sodre, L., J., & Storrie-Lombardi, L. J. 1992, MNRAS, 259, 8P
Tomita, T. M., Browne, J., Shen, C., et al. 2020, Journal of machine learning research, 21
Trump, J. R., Impey, C. D., Kelly, B. C., et al. 2011, ApJ, 733, 60
Trump, J. R., Sun, M., Zeimann, G. R., et al. 2015, ApJ, 811, 26
Veilleux, S., & Osterbrock, D. E. 1987, ApJS, 63, 295
Way, M. J., & Klose, C. D. 2012, PASP, 124, 274
Xue, Y. Q., Luo, B., Brandt, W. N., et al. 2016, ApJS, 224, 15
Zhang, Y., Tu, Y., Zhao, Y., & Tian, H. 2019, in Astronomical Society of the Pacific Conference Series, Vol. 521, Astronomical Data Analysis Software and Systems XXVI, ed. M. Molinaro, K. Shortridge, & F. Pasian, 417