Identifying the Presence of RFI in Visibility Waterfall Plots using Supervised Learning

Jing Liu
PHYS 396 Final Report
Supervisor: Adrian Liu
McGill University Department of Physics
April 26, 2020

Abstract

Identifying irregularities is a crucial step in any data processing pipeline. The Hydrogen Epoch of Reionization Array radio interferometer, located in South Africa, searches for 21-cm emission from the Epoch of Reionization. It returns complex visibilities as waterfall plots of local sidereal time vs. frequency, which may be contaminated with radio frequency interference (RFI). The goal of this project is to create a model to identify the presence of RFI in plots before they are sent further down the processing pipeline. Using supervised learning in a three-layer multilayer perceptron running on simulated data, an accuracy of 0.997 (number of samples $n = 10$, standard deviation $s = 0.006$) was achieved. This result confirms the possibility of binary RFI identification using simple machine learning techniques, and may help reduce the number of plots on which current RFI-flagging algorithms must be run.

1 Introduction

The Hydrogen Epoch of Reionization Array (HERA) is a radio interferometer located in South Africa. HERA is part of an international effort to map the universe during the Epoch of Reionization using 21-cm cosmology techniques [1] [2]. During this period and the Dark Ages that preceded it (more details in Appendix A.1), neutral hydrogen emitted photons as its spin states underwent changes, known as 21-cm emission [3]. This radiation could provide a substantial basis for understanding facets of the early universe such as the properties of the first galaxies and the evolution of large-scale structure [1].

Data from HERA is returned as visibility waterfall plots of local sidereal time (LST) vs. frequency. Visibility is a complex quantity that is essentially a Fourier transform of the brightness of the sky [4]. (A more in-depth treatment of radio astronomy and HERA can be found in Appendix A.2.) In practice, HERA data may be contaminated by sources such as radio frequency interference (RFI), a term applied to transmissions within the observed frequency band from non-celestial sources [5]. These irregularities must be accounted for in order to make use of the data.

Currently, machine learning models are able to delineate RFI directly in waterfall plots [6]. A model that simply identifies the presence of RFI instead could be useful as an earlier step in

the data processing pipeline. To that end, for my PHYS 396 research course[1] with Dr. Adrian Liu[2] this semester, I developed a three-layer multilayer perceptron (MLP) that provides an RFI-identification mechanism. The concepts behind this implementation are introduced in Appendix A.3 (supervised learning) and A.4 (neural networks). This report aims to journal my process and summarize the results and lessons learned from this project.

2 Process

2.1 Creating a First Model

The dataset for this project is composed of waterfall plots generated from `hera_sim` [3]. I made a simple LST vs. frequency plot with three layers: diffuse foregrounds, which are emissions from within the Milky Way; point-source foregrounds, which are mostly extragalactic; and noise [7]. An example visibility plot is shown in Figure 1, with default dimensions of 4000 radians by 1024 MHz. I created a function to layer one visibility plot with m random RFI patterns, then split each version into l parts, creating a total of $l(m+1)$ data points. (To be clear: the term *data point* refers to a piece of a three-dimensional waterfall plot.) Each data point is individually standardized: the mean of its values is set to 0 and the variance to 1.

Using initial values of $m = 5$ and $l = 200$, I created a 1200-point dataset (200 clean, 1000 RFI). Each data point has shape $(20, 2048)$: 20 is the result of dividing the LST range by 200, while 2048 comes from concatenating the extracted real and imaginary components of the data. After experimenting with larger dataset sizes (6600 and 18600), and varying the proportion of clean to RFI data, I eventually found that a balanced 2400-point dataset provides a good compromise between runtime and performance. I did not experiment with changing l and m for a given dataset size; this could be worth considering in the future.

I constructed my first MLP in TensorFlow Keras with three dense layers. The layers and parameters of the model are shown in detail in Figure 2. I used a stratified[4] test set with size 0.33, ReLU activations in the hidden layers, a sigmoid activation in the output layer, and an Adam optimizer with a binary cross-entropy loss. The model was fit over 150 epochs with a batch size of 32. These parameters were initially chosen based on an online tutorial as well as other sources [8].

This first model, along with a version modified to have 25 nodes instead of 10 in the first dense layer–which I will call my second model–achieved a high accuracy, suggesting that the dataset was not difficult to classify. Nonetheless, I continued to experiment with different models and parameters to see how performance would be affected. For example, I implemented stratified k-fold cross-validation, a technique that splits the dataset into k parts and through k iterations, assigns each part to be the test set and all others to be the training set [9]. The advantage of cross-validation is that the model has access to more training data, and is thus more generalizable to variable datasets [9]. However, the accuracy I

[1] https://www.mcgill.ca/study/2019-2020/courses/phys-396

[2] McGill University Department of Physics, Montreal, QC

[3] `hera_sim` (https://github.com/HERA-Team/hera_sim) is a software package that simulates HERA operations and data, taking into account everything from antenna arrangement to data contamination.

[4] The proportion of classes in the training set and test set matches the proportion of classes in the dataset as a whole.

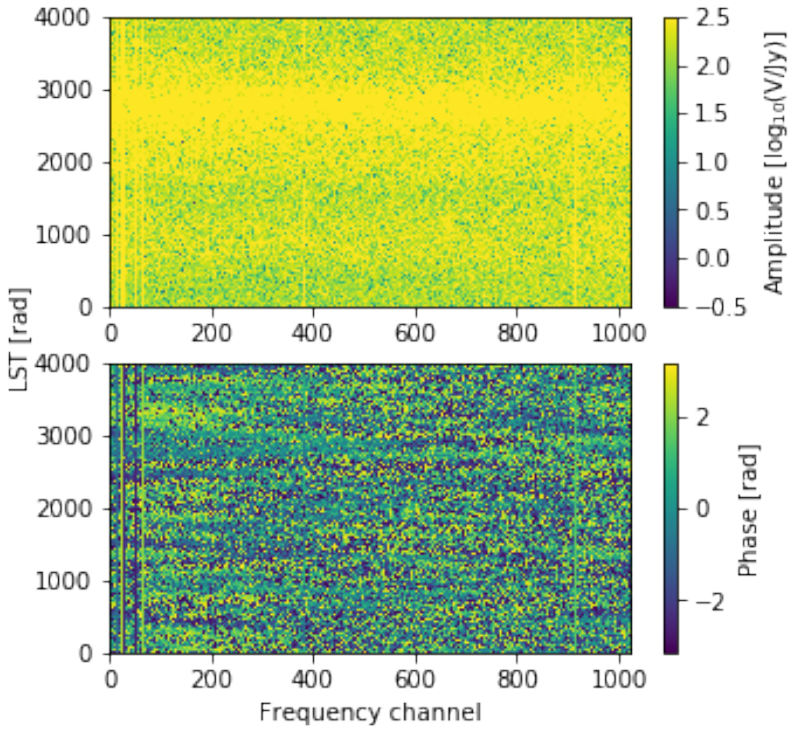

Figure 1: Example waterfall plots, with diffuse and point source foregrounds, RFI, and noise. The same data is shown in both plots: the top plot shows a base-10 log of the absolute value of the data, while the bottom plot shows the phase angle of the data. Note the RFI lines visible near 0 and 900 MHz.

obtained (reported below) was not significantly different than the second model, so I decided to simply continue with the second model.

2.2 Plotting Learning Curves

A primary concern for any machine learning model whether it fits the underlying data appropriately. Underfitting is a scenario where the model disregards much of the data and fails to learn the training set [10]. Overfitting is the opposite situation: the model learns the training data *too* well, having incorporated noise and random errors [11]. Such a model usually performs well on the training set, but poorly on new data [10]. Somewhere in the middle is the elusive "good fit"–having a model that accurately characterizes the true relationship embedded in the data [10].

While my model's high test accuracy suggested a good fit, I decided to verify this using learning curves, which plot a model's loss over time [11]. My model uses binary cross-entropy loss, which can be given by

$$L = -\frac{1}{N}\sum_{i=1}^{N}\left[y_i \cdot \log\left(p(y_i)\right) + (1-y_i)\cdot \log\left(1-p(y_i)\right)\right], \tag{1}$$

where y_i is a label and p is a probability [12]. $p(y_i)$ therefore is the model's predicted

```
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
flatten_3 (Flatten)          (None, 20480)             0
_________________________________________________________________
dense_9 (Dense)              (None, 10)                204810
_________________________________________________________________
dense_10 (Dense)             (None, 8)                 88
_________________________________________________________________
dense_11 (Dense)             (None, 1)                 9
=================================================================
Total params: 204,907
Trainable params: 204,907
Non-trainable params: 0
```

Figure 2: A textual representation of my initial MLP, generated using Keras' `model.summary()`.

probability of the current data point having label y_i. In general, the loss function increases as the predicted probability of the true class decreases.

We usually look at the learning curves of two metrics: the training loss, the model's performance on the training data; and the validation loss, its performance on the validation set [11]. In a model with a good fit, both losses should move together, monotonically decreasing before plateauing at a minimum value [11] (see Figure 3).

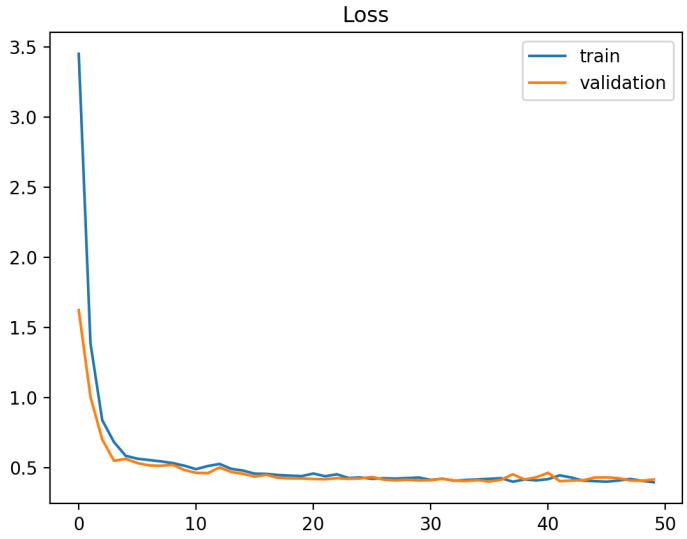

Figure 3: Example learning curves for a model with a good fit [11].

After plotting learning curves for my second model, I observed significant deviations from the ideal curves, detailed below. My attempts to rectify them resulted in a new model with the highest accuracy yet.

3 Results and Discussion

Using my initial 1200-point set (200 clean, 1000 RFI), my first model achieved an accuracy of approximately 0.90. After increasing the number of nodes in the first dense layer from 10 to 25, I obtained a much higher accuracy ($n = 10$) of 0.9865 ($s = 0.005$). Using stratified k-fold cross validation with $k = 3$, an accuracy ($n = 10$) of 0.9848 ($s = 0.009$) was achieved, while with $k = 10$, an accuracy ($n = 1$) of 0.980 ($s = 0.02$) was achieved. Obviously, more test runs would be beneficial, but cross-validation seemed to give similar results to the second models.

Using a new, balanced 1200-point dataset, the second model (tweaked to have test size 0.2 and validation size 0.2) achieved an accuracy ($n = 4$) of 0.987 ($s = 0.005$). Its learning curves (Figure 4) however revealed abnormalities: the validation loss is significantly lower than the training loss in certain sections, which should not occur if the validation set is representative of the training set. In other words, the validation loss does not follow the training loss. Additionally, both losses experience periods of increase. This is problematic since a model that is learning properly should generally improve with every passing epoch.

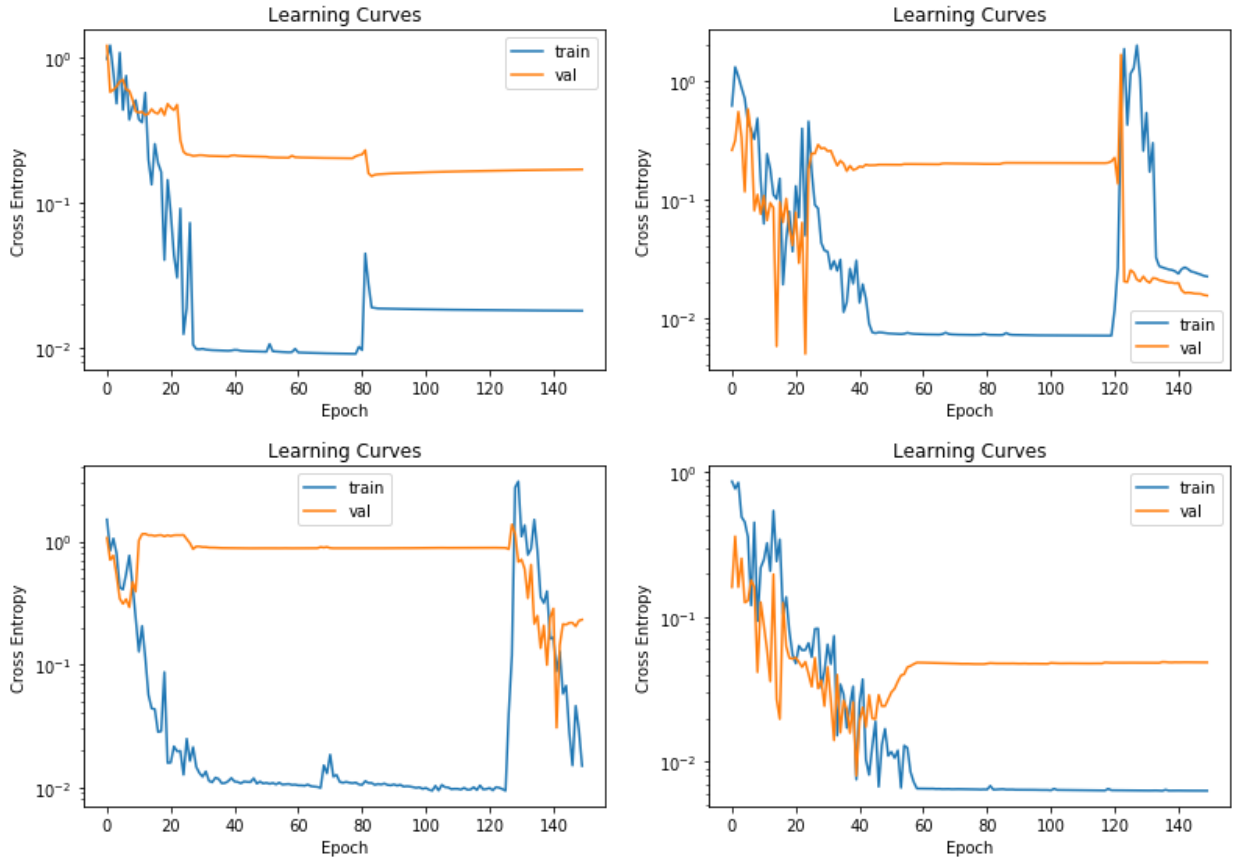

Figure 4: Learning curves for my second model over four runs on a balanced 1200-point dataset. The curves display issues, specified in the main text.

I applied several techniques designed to reduce overfitting to my model. To address the problem of both losses increasing, I first attempted to fix the ordering of the data throughout

all epochs; this ensures that the model doesn't receive a random shuffle of the data at each epoch, thus impeding its progress [13]. However, this did not work as expected and the learning curves still displayed the same issue (Figure 7 in Appendix C). I then implemented a dropout layer, a technique where a proportion (here 0.4) of randomly chosen neurons is discarded in order to discourage reliance on particular neurons [14]. Dropout provided encouraging results: the resulting learning curves showed greater patterns of decrease, though there were still some periods of increase (Figure 8 in Appendix C).

The issue was finally resolved by reducing the model to three layers (excluding the output layer): one dense, one dropout, and one flatten. Figure 5 shows the final model and its parameters, while Figure 6 shows its learning curves during a particular run. The model achieved an accuracy ($n = 10$) of 0.997 ($s = 0.006$).

That the accuracy increased by simplifying the second model suggests that it was too complex for this dataset. The high initial accuracy also indicates that the dataset may be intrinsically easy to classify. That being said, it was worth plotting learning curves because that led to the creation of a better-performing model. In addition, by examining the final model's learning curves, I am more confident that it fits the data well, and as such may prove more useful for new data.

```
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
dense_2 (Dense)              (None, 20, 16)            32784
_________________________________________________________________
dropout_1 (Dropout)          (None, 20, 16)            0
_________________________________________________________________
flatten_1 (Flatten)          (None, 320)               0
_________________________________________________________________
dense_3 (Dense)              (None, 1)                 321
=================================================================
Total params: 33,105
Trainable params: 33,105
Non-trainable params: 0
```

Figure 5: A textual representation of my final MLP.

4 Conclusions

My final model demonstrates that it is possible for a simple MLP to achieve a significant accuracy in identifying in identifying the presence of RFI in simulated visibility plots (0.997, $n = 10$ and $s = 0.006$). Such a model may prove useful in HERA's data processing pipeline as it could reduce the number of data points on which current RFI-flagging algorithms must be run.

The work I accomplished this semester only scratches the surface of RFI identification. There are additional sources of contamination in `hera_sim` that I did not incorporate into the dataset, such as gain and crosstalk. It would also be interesting to revisit cross-validation and observe its effects on the final model's accuracy. Further, it remains to be seen whether my model is successful on real HERA data. Since real data is most likely more complex than my simulated data, newer techniques such as convolutional neural networks, a class of deep

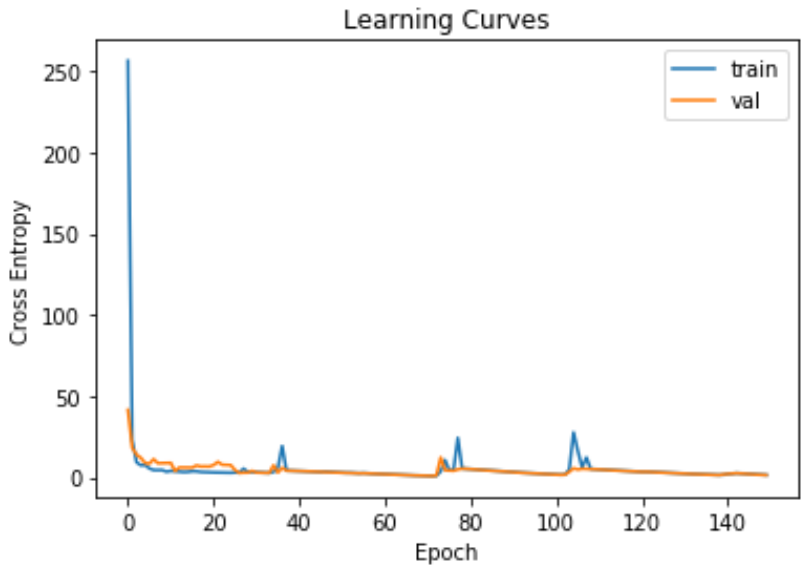

Figure 6: Learning curves for my final model during one run on a balanced 2400-point dataset.

neural networks that is extremely effective for image classification, may be the logical next step [15].

References

[1] D. R. DeBoer, A. R. Parsons, J. E. Aguirre, P. Alexander, Z. S. Ali, A. P. Beardsley, G. Bernardi, J. D. Bowman, R. F. Bradley, C. L. Carilli *et al.*, "Hydrogen epoch of reionization array (HERA)," *Publications of the Astronomical Society of the Pacific*, vol. 129, no. 974, p. 045001, 2017. 1

[2] "HERA Hydrogen Epoch of Reionisation Array radio telescope." [Online]. Available: https://www.sarao.ac.za/science-engineering/hera/ 1, 10

[3] "HI line." [Online]. Available: https://www.cv.nrao.edu/course/astr534/HILine.html 1, 10

[4] Information@eso.org, "Interferometry." [Online]. Available: https://www.eso.org/public/teles-instr/technology/interferometry/ 1, 10

[5] "CASA Radio Analysis Workshop – Caltech, 19-20 January 2012." [Online]. Available: https://science.nrao.edu/opportunities/courses/casa-caltech-winter2012/Isella_Radio_Interferometry_Basics_Caltech2012.pdf 1, 10

[6] J. Kerrigan, P. L. Plante, S. Kohn, J. C. Pober, J. Aguirre, Z. Abdurashidova, P. Alexander, Z. S. Ali, Y. Balfour, A. P. Beardsley *et al.*, "Optimizing sparse RFI prediction using deep learning," *Monthly Notices of the Royal Astronomical Society*, vol. 488, no. 2, pp. 2605–2615, 2019. 1

[7] "LAMBDA - foreground overview." [Online]. Available: https://lambda.gsfc.nasa.gov/product/foreground/index.cfm 2

[8] J. Brownlee, "Tensorflow 2 tutorial: Get started in deep learning with tf.keras," Feb 2020. [Online]. Available: https://machinelearningmastery.com/tensorflow-tutorial-deep-learning-with-tf-keras/ 2, 11

[9] ——, "A gentle introduction to k-fold cross-validation," Aug 2019. [Online]. Available: https://machinelearningmastery.com/k-fold-cross-validation/ 2

[10] W. Koehrsen, "Overfitting vs. underfitting: A complete example," Jan 2018. [Online]. Available: https://towardsdatascience.com/overfitting-vs-underfitting-a-complete-example-d05dd7e19765 3

[11] J. Brownlee, "How to use learning curves to diagnose machine learning model performance," Aug 2019. [Online]. Available: https://machinelearningmastery.com/learning-curves-for-diagnosing-machine-learning-model-performance/ 3, 4

[12] D. Godoy, "Understanding binary cross-entropy / log loss: a visual explanation," Feb 2019. [Online]. Available: https://towardsdatascience.com/understanding-binary-cross-entropy-log-loss-a-visual-explanation-a3ac6025181a 3

[13] G. F. Volpi, "6 amateur mistakes i've made working with train-test splits," Aug 2019. [Online]. Available: https://towardsdatascience.com/6-amateur-mistakes-ive-made-working-with-train-test-splits-916fabb421bb 6

[14] J. Brownlee, "A gentle introduction to dropout for regularizing deep neural networks," Aug 2019. [Online]. Available: https://machinelearningmastery.com/dropout-for-regularizing-deep-neural-networks/ 6

[15] R. Grosse, "CSC 321 intro to neural networks." [Online]. Available: https://www.cs.toronto.edu/~rgrosse/courses/csc321_2018/ 7, 11

[16] M. Haynes, "Recombination era." [Online]. Available: http://hosting.astro.cornell.edu/academics/courses/astro201/recombination.htm 10

[17] "WMAP Big Bang CMB test." [Online]. Available: https://wmap.gsfc.nasa.gov/universe/bb_tests_cmb.html 10

[18] A. Loeb, "The dark ages of the universe," *Scientific American*, vol. 295, pp. 46–53, 12 2006. 10

[19] "MIT Haystack UREI - Undergraduate Research Educational Initiative." [Online]. Available: https://www.haystack.mit.edu/edu/undergrad/materials/tut3.html 10

[20] J. Brownlee, "Supervised and unsupervised machine learning algorithms," Aug 2019. [Online]. Available: https://machinelearningmastery.com/supervised-and-unsupervised-machine-learning-algorithms/ 10

[21] ——, "What is the difference between test and validation datasets?" Jul 2017. [Online]. Available: https://machinelearningmastery.com/difference-test-validation-datasets/ 11

[22] "CS231n convolutional neural networks for visual recognition." [Online]. Available: https://cs231n.github.io/linear-classify/ 11

[23] "Image classification." [Online]. Available: https://cs231n.github.io/classification/ 11

A Background Knowledge

A.1 The Dark Ages and Reionization

About 370,000 years after the Big Bang, as the universe expanded and cooled, electrons and protons that had been ionized came together to form the first neutral atoms in a process called recombination [16]. During recombination, electrons also emitted photons when transitioning to a lower energy state. These decoupled photons, redshifted into radio waves over billions of years, form the cosmic microwave background (CMB) that permeates all space today [17].

After recombination came the period known as the Dark Ages, which lasted until 1 billion years after the Big Bang. The universe had cooled sufficiently for light to travel long distances, but there were only two sources of photons: the aforementioned CMB, and 21-cm line emission from neutral hydrogen due to changes in its spin states [18] [3]. A few hundred millions years later, the first stars, galaxies, and quasars formed, marking the end of the Dark Ages [18]. They emitted radiation and ionized the hydrogen [18]. This process of reionization completed with the emergence of large galaxies, and the universe became the way we see it today [18].

While the period between the CMB's release and the completion of reionization is currently without observation, astronomers are probing 21-cm emission in an attempt to create a map of the sky during this era [18].

A.2 Radio Interferometry and HERA

Compared to its cousin optical astronomy, radio astronomy reveals a parallel sky seen not through visible light, but radio waves [19]. While optical telescopes generally observe light sources such as stars, radio telescopes capture emission from gases, including the 21-cm line mentioned above [19].

HERA is a type of radio telescope known as an interferometer [2]. Because radio waves are longer than visible light waves, for a given diameter, a radio telescope will have a lower resolution than an optical telescope [4]. Interferometry provides a solution: two antennas a certain distance (called a baseline) apart are pointed at the same source. The antennas measure visibility, a complex quantity that quantifies the strength of a Fourier mode of the sky [5]. Their signals are combined in a way that mimics the operation of a telescope dish with a diameter equivalent to the baseline, creating a higher resolution visibility [19].

Each baseline measures a certain Fourier mode, and by combining different baselines (as well as variations in individual baselines, as the Earth rotates) we can map out the sky at all frequencies [4] [19]. HERA returns these visibilities as waterfall plots, which show the desired quantity (in our case visibility) as a function of two parameters (time vs. frequency).

A.3 Supervised Learning

Given a waterfall plot from HERA, we want to determine whether it is clean or has RFI. This binary classification problem can be tackled via a form of machine learning called supervised learning: the model or classifier is fed a training set, an input consisting of waterfall plots labelled as one of two classes: clean or RFI [20]. These labels result from *a priori* knowledge

and are considered ground truth values. A portion of this set is split into a validation set, used to measure the model's performance [21]. The classifier uses the training set to learn the distinguishing features of each class in the dataset as it iteratively makes predictions on the validation set's labels; its accuracy in doing so (i.e. the proportion of correctly-predicted labels) is evaluated and informs the next iteration of learning [21]. Accuracy is related to what's known as a loss function, which quantifies the error in predicting the correct labels; there are many loss functions with different implementations, but in general, for a well-fit model, accuracy increases as loss decreases [22] [8]. Therefore the goal is to minimize loss, which generally happens with every epoch (pass through the entire training set) [23].

After the model has finished learning, it is evaluated on the test set, a previously unseen, labelled dataset [23]. Given an appropriate test set, the model's performance on the test set should predict its performance on data for which there are no labels–namely, real data from HERA.

A.4 Neural Networks

One method of performing supervised learning involves creating artificial neural networks inspired by the brain: these are systems comprising layers of many individual units called neurons [15]. In a feed-forward neural network, information flows in only one direction [15]. A neuron receives inputs x_i from connected neurons in the previous layer; these connections have strengths associated to them, called weights w_i [15]. The neuron sums over its inputs multiplied by their weights and adds a bias term b to account for an absence of inputs [15]. Then, it applies a nonlinear function called an activation function f to this sum [15]. This activation function determines whether the neuron will return a value y to its successors [15]. The entire process can be expressed as $y = f(\sum_i x_i w_i + b)$ [15].

A multilayer perceptron is a type of feed-forward network in which every neuron in a layer is connected to every neuron in the previous layer as well as the following layer; in other words, the network is fully-connected [15]. Information is processed in increasing levels of abstraction, and features, or distinguishing characteristics in the data, are discerned [15]. Simple problems such as this project can be solved with only one or two layers, but more complex problems (e.g. more difficult image classification, or classifying parts of an image) can benefit from many additional layers.

B Code

The code for this project can be found at https://github.com/musicbyjing/cosmic_dawn_rfi.

C Supplementary Figures

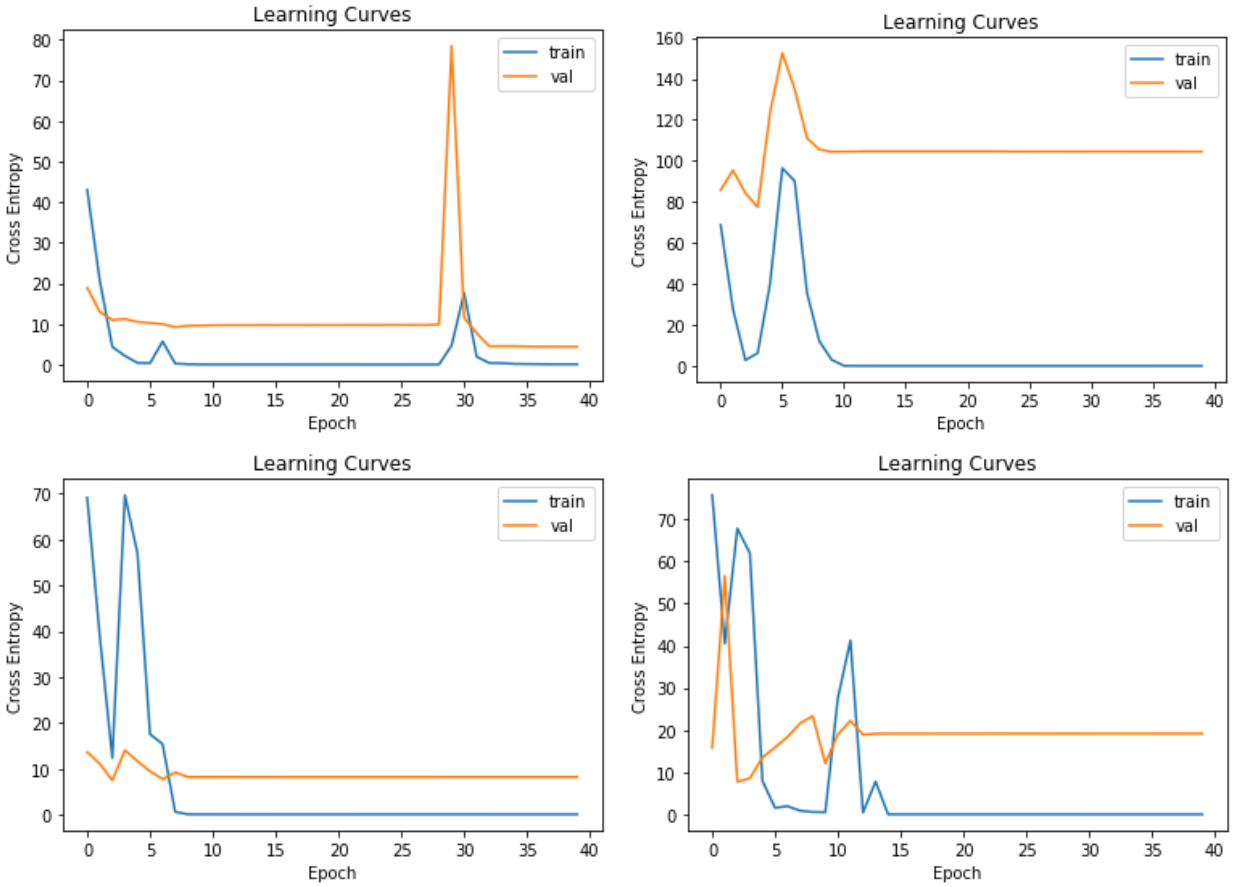

Figure 7: Learning curves for my second model over four runs on a 6600-point dataset (600 clean, 6000 RFI). Shuffling of the data between epochs has been deactivated. An accuracy ($n = 4$) of 0.984 ($s = 0.009$) was achieved.

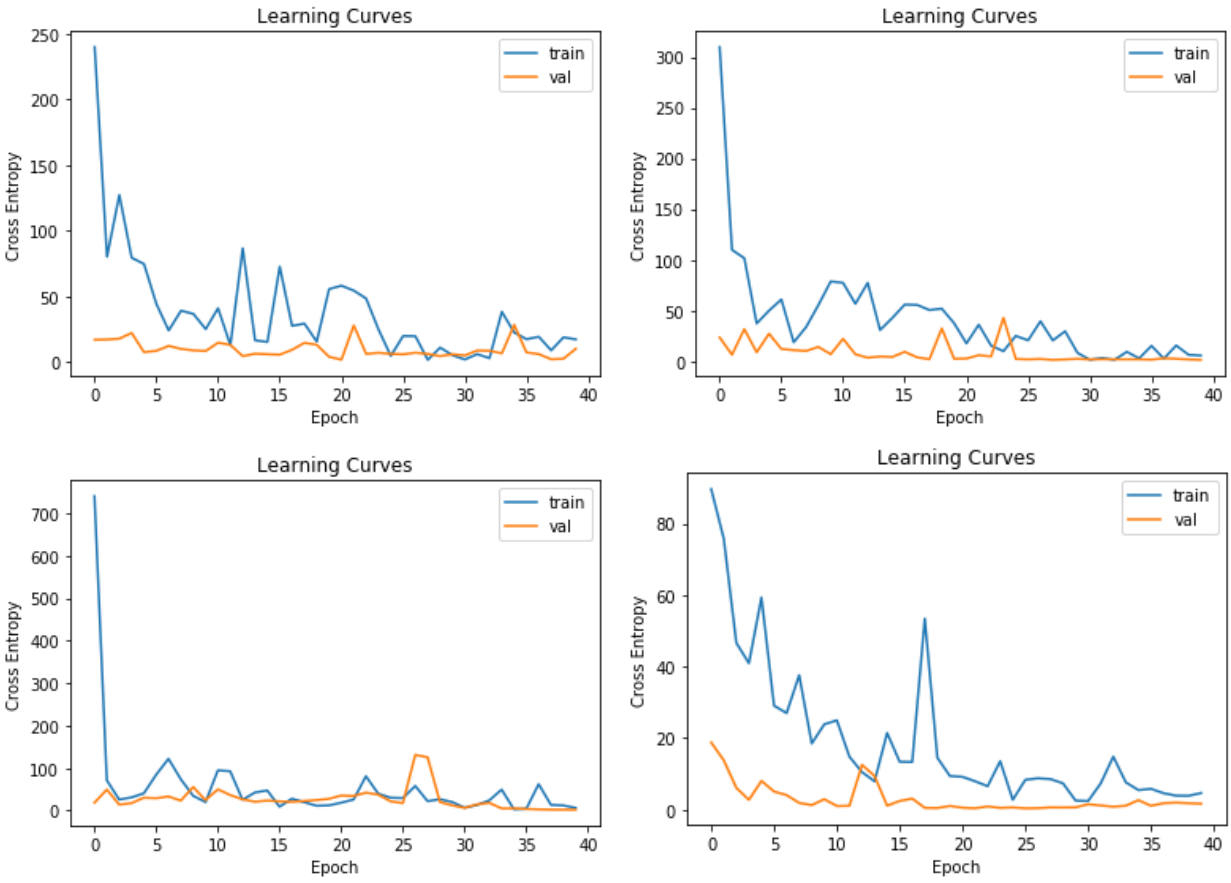

Figure 8: Learning curves for my second model over four runs on a 6600-point dataset (600 clean, 6000 RFI). A dropout layer of 0.4 was applied following the input layer. An accuracy ($n = 4$) of 0.995 ($s = 0.001$) was achieved.