The first deep-learning search for radio technosignatures from 820 nearby stars

Peter Xiangyuan Ma[1,2,3*], **Cherry Ng**[3,4,5], **Leandro Rizk**[6], **Steve Croft**[4,5], **Andrew P. V. Siemion**[4,5,7,8], **Bryan Brzycki**[4], **Daniel Czech**[4], **Jamie Drew**[9], **Vishal Gajjar**[4], **John Hoang**[4], **Howard Isaacson**[4, 10], **Matt Lebofsky**[4], **David MacMahon**[4], **Imke de Pater**[4], **Danny C. Price**[11,4], **Sofia Z. Sheikh**[4], **and S. Pete Worden**[9]

[1]Department of Mathematics, University of Toronto, 40 St. George Street, Toronto, ON M5S 2E4, Canada
[2]Department of Physics, University of Toronto, 60 St. George Street, Toronto, ON M5S 1A7, Canada
[3]Dunlap Institute for Astronomy & Astrophysics, University of Toronto, 50 St. George Street, Toronto, ON M5S 3H4, Canada
[4]Department of Astronomy, 501 Campbell Hall, University of California, Berkeley, CA 94720, USA
[5]SETI Institute, Mountain View, CA 94043, USA
[6]David A. Dunlap Department of Astronomy & Astrophysics, University of Toronto, 50 St. George Street, Toronto, ON M5S 3H4, Canada
[7]Department of Astrophysics/IMAPP,Radboud University, Nijmegen, Netherlands
[8]University of Malta, Institute of Space Sciences and Astronomy
[9]Breakthrough Initiatives, Moffett Field, CA 94035, USA
[10]Centre for Astrophysics, University of Southern Queensland, Toowoomba, QLD, Australia
[11]International Centre for Radio Astronomy Research, Curtin University, Bentley WA 6102, Australia
[*]peterxy.ma@mail.utoronto.ca

ABSTRACT

The goal of the Search for Extraterrestrial Intelligence (SETI) is to quantify the prevalence of technological life beyond Earth via their “technosignatures". One theorized technosignature are narrowband Doppler drifting radio signals. The principal challenge in conducting SETI in the radio domain is developing a generalized technique to reject human radio frequency interference (RFI) that dominate the features across the band in searches for technosignatures. Here, we present the first comprehensive deep-learning based technosignature search to date, returning 8 promising ETI signals-of-interest for re-observation as part of the Breakthrough Listen initiative. The search comprises 820 unique targets observed with the Robert C. Byrd Green Bank Telescope, totaling over 480 hr of on-sky data. We implement a novel $\beta-$Convolutional Variational Autoencoder with an embedded discriminator combined with Random Forest Decision Trees to classify technosignature candidates in a semi-unsupervised manner. We compare our results with prior classical techniques on the same dataset and conclude that our algorithm returns more convincing and novel signals-of-interest with a manageable false positive rate. This new approach presents itself as a leading solution in accelerating SETI and other transient research into the age of data-driven astronomy.

1 Introduction

“Are we alone?” is one of the most profound scientific questions humans have asked. The search for extraterrestrial intelligence (SETI) aims to answer this question by looking for evidence of intelligent life elsewhere in the galaxy via the “technosignatures” created by their technology. The majority of technosignature searches to date have been conducted at radio frequencies, given the ease of propagation of radio signals through interstellar space[1], as well as the relative efficiency of construction of powerful radio transmitters and receivers. Radio technosignature candidates, whether intentional transmissions or radio “leakage”, are in principal easily distinguishable from natural astrophysical radio emissions. For example, they might be narrow-band (on the order of 1 Hz) and / or exhibit Doppler drifts due to the relative motions of transmitter and receiver[2]. The detection of an unambiguous technosignature would demonstrate the existence of extraterrestrial intelligence (ETI) and is thus of acute interest both to scientists and the general public.

The first modern technosignature search, Project Ozma, was conducted in 1960[3] where 400 kHz of bandwidth were searched from the observation of two stars. Subsequent all-sky searches such as[4] surveyed a much larger region of the sky. Although state-of-the-art at the time, by today’s standard the sensitivity of these studies was limited, only capable of detecting an Arecibo-like transmitter to $\sim$5 pc from Earth. Radio SETI has expanded exponentially even in just the last decade by

almost every metric, including the number of stars searched, the frequency ranges covered, and the sensitivity of the telescope instruments employed. Currently, one of the main driving forces of SETI research is the Breakthrough Listen Initiative[1]. One of the mandates of Breakthrough Listen is to survey 1 million nearby stars to conduct the most comprehensive technosignature search to date. Since 2016, Breakthrough Listen has been using the Robert C. Byrd Green Bank Telescope (GBT) in the U.S. and the Parkes "Murriyang" Telescope in Australia to search thousands of stars and hundreds of galaxies across multiple bands for technosignatures[5–8]. Breakthrough Listen's projects on GBT and Parkes have resulted in the aggregation of over 20 PB of observational data, including over 2 PB in a growing public archive[29].

Despite the fact that many radio telescopes used for SETI are located in radio-quiet zones, Radio Frequency Interference (RFI) due to human technology still poses a major challenge for SETI research. In order to reject RFI, one of the techniques employed by the Breakthrough Listen team is that of spatial filtering using "cadence" observations, also known as "position switching". The key idea is that every primary target observation is interspersed with observations of nearby targets as "OFF-source" scans. The ON-OFF pattern is repeated three times to improve RFI rejection. If a signal of interest detected in the observation were human-made and entering the telescope receiver system through the sidelobes of the antenna, the signal would appear in multiple adjacent observations within a cadence irrespective of whether it is on or off source due to the near-field nature of RFI. In contrast, an ETI signal observed on-axis in the primary beam should only appear in the "ON-source" scans and not in any other "OFF-source" scans in the cadence set. One of the most extensively used SETI algorithms in Breakthrough Listen's analysis is TURBOSETI[35,10], which is designed to detect these narrowband drifting signals by implementing the "tree de-Doppler" algorithm for incoherent Doppler acceleration searches described by[11]. The TURBOSETI package also takes into account the cadence filtering criteria. However, one can imagine a nearly infinite range of possible ETI signals which might not be captured by the de-Doppler algorithm. Similarly, the presence of RFI in observational data can often result in a high false positive rate, as shown by previous searches using TURBOSETI on similar GBT dataset as employed in this work, with 29 million hits reported by[5] and 37 million hits reported by[7].

Recently, Machine Learning (ML) has seen an increasing application in the field of astronomy. One example is the use of Convolutional Neural Networks (CNN)[12] for tasks like galaxy / star classifications, which have accelerated research in this field to tackle the ever growing size of data collected by new survey instruments[13]. ML has also previously been applied in the context of SETI work, for example in a generic signal classifier for observations obtained at the Allen Telescope Array[14], a CNN-based RFI identifier[15], as well as anomaly detection algorithms[16,17]. In the latter, a stacked Convolutional LSTM (Long Short Term Memory; see, e.g.,[18]) Generative Adversarial Network (ConvLSTM-GAN) was used to train a model which attempts to generate the next observation in a cadence. If there is an anomaly, the model will fail to create the anomaly since the model was trained on recreating what is "typically" expected in the next observation. Although the model performs well, the use case is somewhat different than the situation presented in this work, as it predicts how anomalous the next time stamp is versus a classification on the entire cadence.

Modern deep learning techniques have the ability to generalize relationships in big datasets which could have major implications in SETI work. One notable subclass of ML algorithms is disentangled learning. Disentangled learning defines a neural network that is able to implicitly learn and identify uncorrelated features within a dataset. For example, if a network was trained on different images of apples that have been rotated or translated, the neural network would be able to identify rotation and translation as key features of the data and can disentangle that knowledge from other elements of the data, such as the shape or color of the apples. This design ultimately tackles the black box problem with neural networks by forcing the network to learn interpretable features from training data[19]. Frameworks like $\beta-$Variational Autoencoders (β-VAE) have demonstrated this disentanglement property[20]. Models like β-VAE present an exciting way to develop deep neural nets such that they learn human-interpretable and physical features from the data. This is of high importance when conducting analysis using deep learning models, as one ought to verify if the model can truly represent real data. More concretely, this design of autoencoders allows the model to implicitly learn the features of "cadence filtering" and "narrowband Doppler drifting signals", meaning that a wider range of potential ETI signals can be searched. Furthermore, autoencoders can also implicitly learn how RFI appears in observational data, without having to be programmed for the nearly-infinite possible morphologies an RFI signal can take. Another relevant development in ML is the use of decision trees and random forests to help classify samples given a selected number of features[21]. Despite the widespread popularity in earlier works in ML, and their eventual replacement with neural networks as the forefront of ML, decision trees and random forests still present themselves as a fast and robust model in classifying data with a small number of features and classes.

Here we apply ML techniques from recent advances in disentangled deep learning in combination with a random forest decision tree to search through a set of data from the GBT L-band receiver, which operates across 1.1–1.9 GHz, as described in Section 2. In Section 3 we present the ML algorithm. In Section 4 we evaluate the performance of our algorithm and in

[1] https://breakthroughinitiatives.org
[2] https://seti.berkeley.edu/opendata
[3] https://github.com/UCBerkeleySETI/turbo_seti

Section 5 we describe the implementation of this analysis pipeline. In Section 6 we present the result of our search. This is followed by a discussion in Section 7 and we conclude in Section 8.

2 The GBT 1.1–1.9 GHz dataset

We have chosen to deploy an ML SETI algorithm on the Breakthrough Listen GBT 1.1–1.9 GHz dataset, sourced from several different observational campaigns, as it is one of the largest homogeneous SETI datasets available in a single location at the Berkeley SETI Research Center. The majority of this dataset has been made publicly available[4]. The dataset used in this work has a frequency range of 1023−1926 MHz and consists of a total of 1004 cadences from 820 unique targets observed over 480 hr. Each cadence has six 4.8-min observations recorded in HDF5 format. In the analysis described here, we exclusively work with the fine-frequency resolution data[9]. More than half of the cadences have a frequency resolution of 2.79 Hz and 322,961,408 frequency channels. The remaining 397 cadences taken before 2016 April have a frequency resolution of 2.84 Hz and 318,230,528 channels per observation. All data have a time resolution of ∼18 s and 16 time bins for each observation within the cadence. Parts of the band are affected by instrumental artifacts, including steep roll-off on the band edges and at the boundaries of an analog notch filter. These regions are at frequencies $\nu < 1.1$ GHz, $\nu > 1.9$ GHz and $1.2 < \nu < 1.34$ GHz. Together these equate to about 30% of the full band and are excluded from our search.

Two types of cadences have been used in the GBT observations, the ABACAD pattern and the ABABAB pattern. The difference between these is that the "ON-source" scans (the A scans) can be interspersed with identical "OFF-source" targets (the B scans) or three different "OFF-source" targets (B, C, D). In both cases, the minimum angular offset between ON and OFF source scans is at least six beamwidths, which is well beyond the primary and side lobes of the GBT beams. In fact, the majority (over 80%) of the cadences analyzed have an angular offset of as much as 38 beamwidths. We treat both types of cadences the same way in our analysis, since for the purpose of spatial filtering we only care that the "OFF-scans" are sufficiently far from the "ON-scans". In total, ∼ 120 TB of data has been analyzed in this work.

Another reason we have chosen to work with this dataset is because a thorough ETI search has previously been conducted on it using a standard de-Doppler technique[5,7]. This provides a way to benchmark our algorithms as we can compare the output of two completely different methods. Out of the 1004 cadences analyzed here, 688 were studied by[5] and 755 were in the sample used by[7]. The small mismatch in the source list is because we have rejected several cadences that have incomplete observations with less than 16 time bins, which are incompatible with our ML algorithm. We have also discarded cadences that have a long (≥2 min) pause between the individual observations, since that would significantly alter the data characteristics due to longer term variation in telescope performance and our ML model has not been trained to detect those.

3 Methodology and Machine Learning Algorithm

3.1 Preprocessing and Training Data

Neural networks perform better if the input data has a shape close to that of a CNN kernel which typically has a 1:1 aspect ratio. In contrast, the fine-resolution GBT data have a very elongated dimension, with over 300 million frequency bins by 16 time bins per observation. Indexing small partitions of the frequency band and searching them independently can solve that problem, but a small frequency band will reduce the maximum drift rates our algorithm is sensitive to, since the algorithm cannot detect any signal that drifts beyond the diagonal of the input data shape. Here we define the maximum drift rate detectable by a search to be when a signal starts drifting from the corner of the first observation and drift out near the opposite corner of the third ON scan, spanning a total of four observations (ON-OFF-ON-OFF) worth of time (see bottom panel of Fig. 1a).

We find a good compromise in using snippets of 4096 frequency bins, which provides sensitivity to a maximum drift rate of 4096×2.8 Hz over 16×18 s per observation over 4 observations $\approx \pm 10$ Hz/s. These higher drift rates have a proportionally higher chance to drift out of at least one of the snippets depending on where the signals start, partially altering the expected ON-OFF pattern and reducing the sensitivity of our model towards these cases. In order to alleviate this potential issue, we devise an overlapping search that will cover snippets offset by 2048 frequency bins which is half the size of a snippet window (Fig. 1). In the scheme illustrated in the top panel of Fig. 1a, any drift rate lower than the diagonal drift rate, i.e. 4096×2.8 Hz/$(6 \times 16 \times 18$ s$) \approx 6$ Hz/s will either be detectable by the blue snippets or by the red overlap search. Only a very small yellow-highlighted region on the far left is missed since too much of the drifting signals happen before the start of the snippet. Compared to the large number of snippets per cadence, this equates to a 99.999% of parameter space coverage. For signals with higher drift rates than the diagonal, the overlap search would not be quite enough to fully cover some of the parameter space. The bottom panel of Fig. 1a shows the most extreme case at the maximum detectable drift rate of 10 Hz/s. At this point, apart from a slightly larger triangle on the left that will be missed, there are now gaps between the blue and the red drifting signals that we are not sensitive to. The probability of having the signal detectable within a cadence drops to ∼75 % at ±10 Hz/s (Fig. 1b).

[4] https://seti.berkeley.edu/opendata

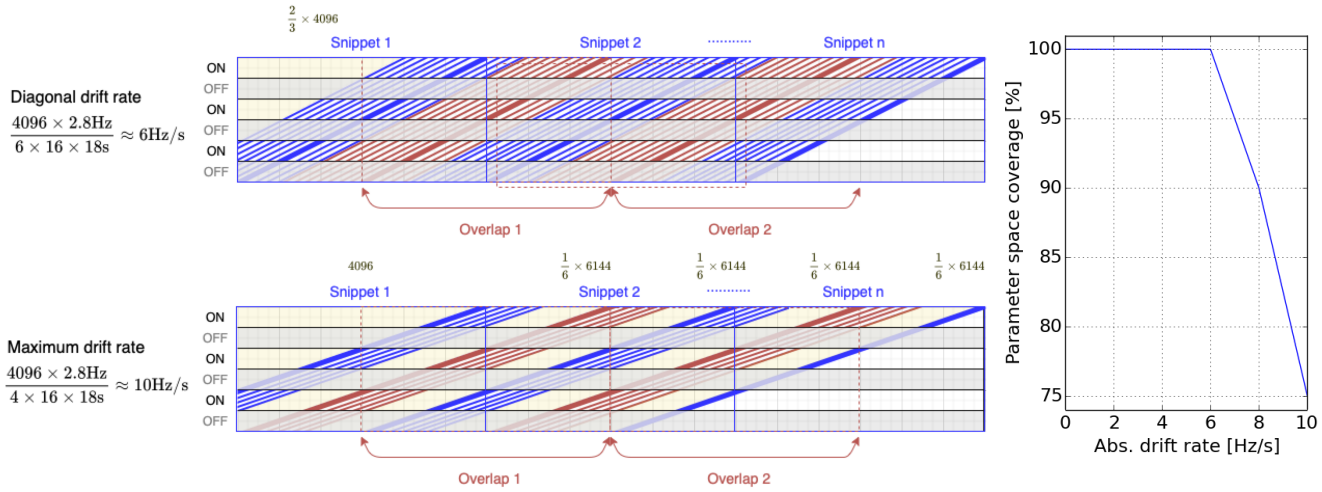


Figure 1. (a) A sketch of the overlap search method. Only positive drifting signals are shown for brevity as the negative drifts are symmetric. The top panel shows the diagonal drift rate at 6 Hz/s. The blue drifting signals are detectable by the regular snippets (blue), whereas the red drifting signals in between are detectable by the overlap search (red). A small triangular region on the far left is not detectable by our search because any drifting signal starting there will have part of its pattern before the start of the first snippet, in the sense that not all three "ON" scans contain a signal. This triangular region exists for all non-zero drift rates and gets progressively larger. But in all cases, this represents a tiny region out of the many snippets within a cadence and thus we have practically fully coverage for any drift rate below the diagonal drift. The bottom panel shows the scenario with the maximum detectable drift rate at 10 Hz/s. For example, this is when the first blue drifted signal barely shows up in the third "ON" scan. In this case, apart from the triangular region on the left, there will be additional gaps throughout the cadence that are not caught by neither the snippets nor the overlap search. The number of frequency channels missed are indicated on top of each of the yellow-highlighted regions. (b) The corresponding parameter space coverage as a function of the absolute drift rate. We have practically full detectability up to the diagonal drift rate of 6 Hz/s. Beyond that, the coverage drops to 75% at the maximum drift rate of 10 Hz/s.

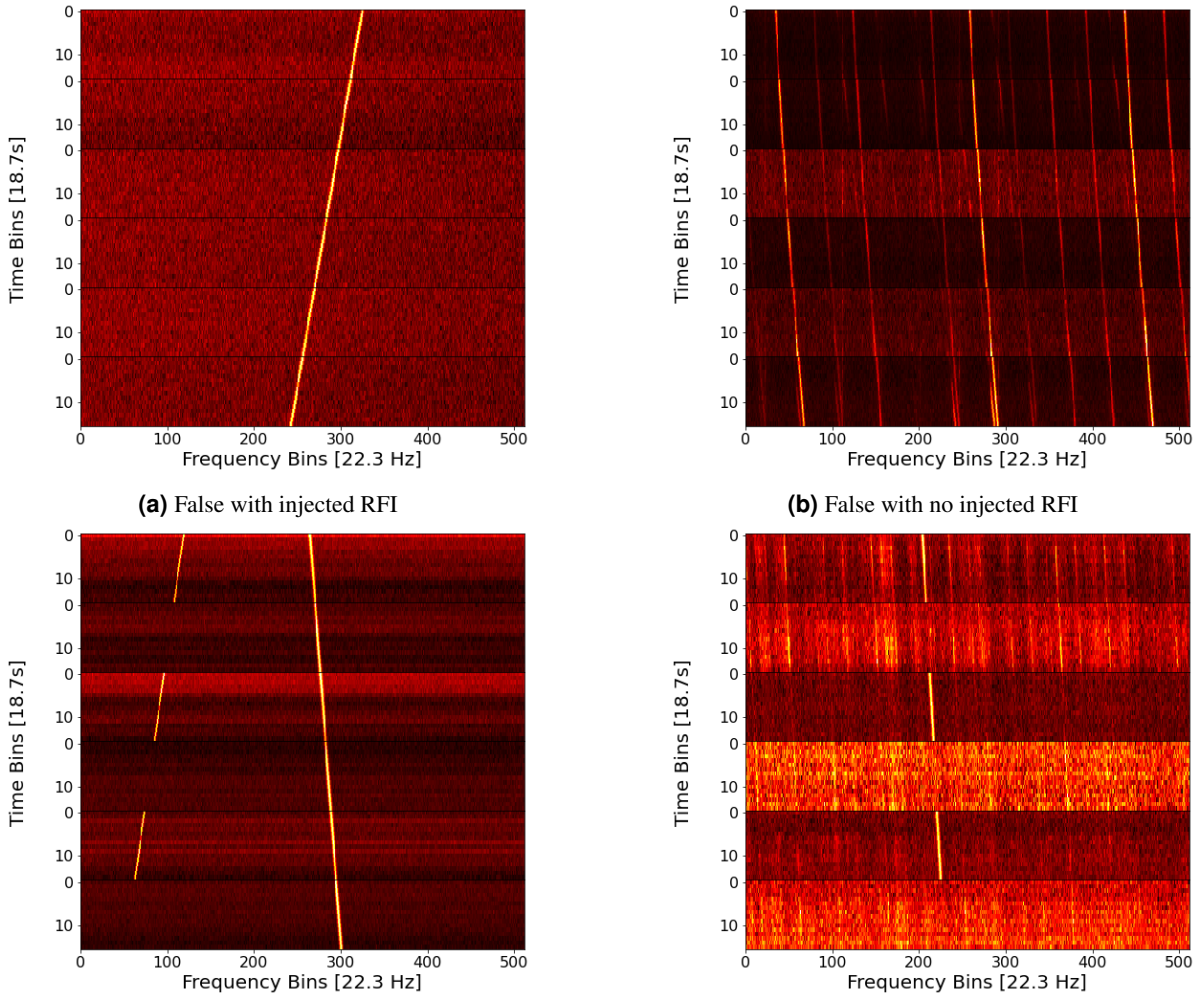
We further downsample the snippets in frequency by a factor of 8 (to a frequency resolution of 22.32 Hz or 22.72 Hz) using a localized mean approach, which results in an input spectrogram with shape [time, frequency]=[16, 512]. This helps to further improving the training time and performance of the neural network, at the expense of decreasing the S/N of a potential signal contained in one channel by $\sqrt{8}$.

In order to train our ML algorithm and evaluate the model, we need to provide labelled data for the model to learn from. There are three main categories of labelled data that are relevant: (1) False data with no ETI signals, (2) True data with ETI signals, and (3) True data with ETI signals and RFI. Since we do not have a sample of real observational ETI signals for this purpose, we generate simulated events by artificially injecting signals into the input spectrograms using the Python package SETIGEN[5]. A total of 14,711 different snippets of backgrounds are used, obtained from three different cadences (see Table 1 for more details). These backgrounds are taken from regions of the band (1418.7 to 1587.9 MHz) that overall showed higher than average RFI as determined by the previous TURBOSETI search[5], which we can confirm by visually inspecting these regions in the three cadences. The intention is that our model is trained with some examples of the most RFI-contaminated observations. A more rigorous selection of backgrounds could be achieved by identifying snippets with statistical anomalies in energy, as well as by employing a larger number of cadences to increase the diversity in the backgrounds. However, we believe that our large sample of backgrounds should provide a good variety of scenarios to help the generalization of the training model.

[5]https://github.com/bbrzycki/setigen

Training target	Cadence	Freq range (MHz)	Bandwidth (MHz)	Freq. res (Hz)	No. of snippets	No. of test samples
	HIP 110750	1418.7−1475.1	56.4	2.79	4928	
β-VAE	HIP 13402	1475.1−1531.5	56.4	2.83	4855	120,000
	HIP 8497	1531.5−1587.9	56.4	2.79	4928	
Random Forest	HIP 110750	1418.7−1475.1	57.139	2.79	4928	24,000

Table 1. The breakdown of number of background snippets and the frequency ranges of each cadence used for the training dataset. The number of test samples simulated in each training set is also listed. The HIP prefix in the source name shows that these targets are drawn from the Hipparcos catalog[22].



(a) False with injected RFI

(b) False with no injected RFI

(c) True with injected RFI

(d) True with no injected RFI

Figure 2. Examples showing the four types of training data. These spectrograms have a frequency resolution of 22.3 Hz and a time resolution of 18.7 s.

From the 14,711 unique backgrounds, we randomly draw a heuristic 120,000 sample to form the training set. For (1), we use one quarter (30,000 samples) of the original backgrounds without injections and an additional 30,000 samples with RFI injected to the backgrounds. Another 30,000 samples are labelled as (2), where we inject ETI signals into the backgrounds and are referred to as "True Single Shot" in this paper. Finally, the last 30,000 samples are labelled as (3), where we inject both ETI signals and RFI with a 1:1 relative intensity ratio into the background spectrograms. We have not simulated all combinations of intensity ratios since the model will in principle generalize this parameter. See Fig. 2 for an example of these three types of labelled data used. A caveat is that if a true, non-synthetic ETI signal is present in the original backgrounds, it would lead to a

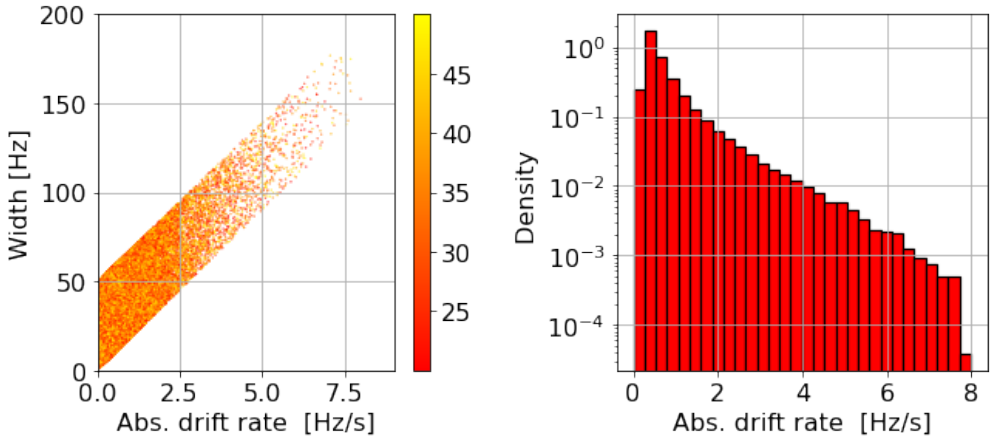


Figure 3. (left) The distribution of widths vs drift rates for a randomly picked sub-set of 5000 synthetic signals generated for the training set. The colors represent the S/N of each signal. There are fewer samples with higher drift rates as we discard everything that drifts partially out of frame. (right) The corresponding density distribution of our simulated sample as a function of absolute drift rate.

mislabelling of our training dataset. This is an unlikely scenario, and our large, 120,000-sample training set should minimize the effect of individual mislabelled data. The only difference between a synthetic RFI and an ETI signal is that the RFI will have the same drifted signal across all six cadences, whereas in the case of a synthetic ETI signal, we replace every adjacent observation with the original, unaltered spectrogram to create the ON-OFF-ON-OFF-ON-OFF pattern.

Another advantage of using simulated data is that we can control the injection parameters, namely the signal-to-noise ratio (S/N), the drift rate and the width of the signal. The S/N in SETIGEN is estimated from the background noise using sigma clipping. We set a range of 10 to 50 for the S/N and a drift rate range of ± 8 Hz/s. Note that simulated ETI signals with zero drift rate are included. For the width (w), we choose a range where $5 + \frac{1}{\text{drift}} \leq w \leq 55 + \frac{1}{\text{drift}}$ Hz. The lower bound offset of 5 Hz is chosen to be slightly wider than the size of a frequency bin which is approximately 3 Hz. This is necessary because signals injected into the spectrogram are quantized to the frequency and time resolutions of the spectrogram. In effect, if there were a highly drifted and narrow signal, it would appear as sparse dots in the spectrogram as opposed to a continuous signal. This would not have been an issue had we injected signal into the raw voltage data from the telescope; storing raw voltage data, however, is extremely memory-intensive, so we are limited to output filterbank data products for this particular set of archival data. Within the above ranges, we randomly draw a combination of S/N, drift rate and width for each simulated signal. See Fig. 3 for the distribution of the final set of injected parameters. Note that there are fewer simulated signals at higher drift rates. This is partly because there is a higher chance for them to drift out of our detectable parameter space as can be seen in Fig. 1, and also because we have not simulated signals that start outside of a snippet but drift into it. Fig. 3b shows the density distribution of our simulated sample as a function of drift rate. The fact that we do not have a balanced test class with respect to drift rate means our model is less well-trained for the high-drift region of the parameter space (see further discussion in Section 7).

In order to obtain the clustering loss matrix and the reconstruction loss vector (see Section 3), we need to train the encoder concurrently with concatenated (collapsed) cadences as well as non-collapsed cadences. We generate separate training sets for these to increase the diversity of the training set — that is, we have 120,000 samples of concatenated cadences as well as an independent 120,000 samples of false cadences and another 120,000 samples of true cadences. Refer to Fig. 4 for the breakdown of the size of the training set. Overall, it takes roughly 12 hr to generate and train the full model with 100 iterations each time on our machines (see Section 5 for further details on the specs of these computers).

The dynamic range of these GBT data can vary by as much as a factor of 10^{11}. After the injection stage, we log normalize the data, shift everything to be positive and we scale the data to have a final range within 0 and 1. Note that the normalization is done per snippet of each cadence and not across the entire cadence.

3.2 The β-VAE ML Architecture

In this analysis we use a VAE-based ML model[19], which is one of two main families of deep generative models that can produce highly realistic versions of the input data. Autoencoders are neural networks architectures with an encoder and a decoder that form a bottleneck for the data to go through. They are trained to minimize the loss of information during the encoding-decoding process, which is achieved by iterations of gradient descent with the goal of diminishing the reconstruction error. A VAE is a

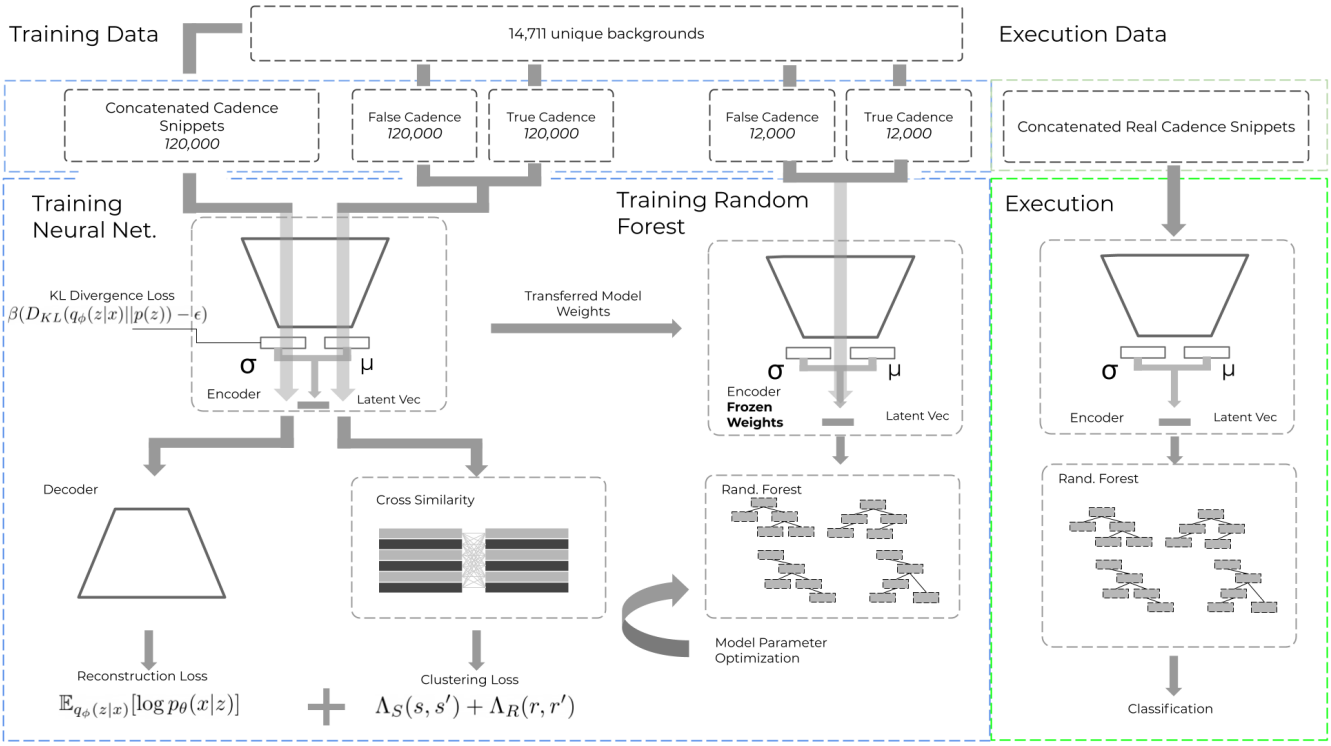


Figure 4. Model training and execution scheme. Backward propagation of the neural network training is not shown for brevity.

special kind of autoencoder whose encoding distribution is regularized. The term "variational" reflects the relationship between the regularization and the variational inference method in statistics.

Our encoder consists of 3×3 kernels in each convolutional layer with a total of 8 convolutional layers with filter sizes of [16, 32, 64, 128][12]. The number of convolutional layers are determined using a Bayesian optimization technique[23], whereas the filter sizes are empirically fine-tuned manually. We first stack these convolutional layers to extract "spatial" features from the spectrogram and subsequently feed the data through a traditional fully-connected neural network. Here the model splits off into two layers: the mean (μ) and the log of the deviation (σ). These two are fed into a sampling layer as part of a standard encoder model[19]. The decoder attempts to recreate the original spectrogram by performing the reverse of what the encoder did, as it is by definition inversely symmetrical to the encoder. The latent space in between the encoder and the decoder has a dense layer of 512 parameters and the latent vector is 8 dimensional in shape. The final model design is shown in Fig. 4. It is implemented using TENSORFLOW[6] and KERAS[7].

The original VAE framework can be augmented with a single hyperparameter β in the loss function that modulates the learning constraints applied to the model, specifically by controlling how much the model is penalized for how the model constructs the sample layer[20]. For our model we empirically select $\beta = 1.5$, which appears to best help the model to learn disentangled features in the latent space (see Section 4.3). A mathematical representation of this is shown by Equation 1, which is a slightly modified version of Equation 3 in[19]. In the referenced paper, $\mathscr{L}$ was used to represent the Lagrangian. Here we flip the sign of the equation to show the loss function L as follow:

$$L(\theta,\phi,\beta;x,z) = \overbrace{-\mathbb{E}_{q_\phi(z|x)}[\log p_\theta(x|z)]}^{\text{Reconstruction Loss}} + \overbrace{\beta(D_{\mathrm{KL}}(q_\phi(z|x)||p(z)))}^{\text{KL Divergence}}. \tag{1}$$

In this Equation, x is the observational data and we want to develop an unsupervised deep generative model which will learn the joint distribution of the data x and a set of generative latent factors z. This effectively is the probabilistic decoder, $p_\theta(x|z)$, with generative model parameters θ, where for a given z it produces a distribution over the possible corresponding values of x. Inversely, $q_\phi(z|x)$ is the probabilistic encoder with recognition model parameters ϕ, where for a given data point x it produces a distribution over the possible values of z from which the data point x could have been generated. $\mathbb{E}_{q_\phi(z|x)}$ computes

[6]https://www.tensorflow.org/
[7]https://keras.io/

the (log-)likelihood of effectively each pixel that the model reconstructs. To summarize, the two main terms in this Equation are the reconstruction loss and the β-weighted Kulback-Leibler (KL) divergence. The KL divergence metric is used to measure the statistical similarities between the mean and the log-normalized standard deviation layers.

Specifically for our analysis, we also want the model to be able to group observations with similar features together in the model's latent space. In order to achieve this we incorporate a clustering loss which we denote Λ. We use Λ_S for SETI and Λ_R for RFI signals. We compute the latent vectors of the cadence for true SETI cases called S with corresponding latent factor s' and an input of s. Similarly, we label the corresponding parameters for false SETI cases (RFI) as R with corresponding latent factor r' and an input of r:

$$S = q_\phi(s'|s)\,, \quad R = q_\phi(r'|r)\,. \tag{2}$$

Each parameter represents a set of six feature vectors corresponding to the six observations each encoded from the cadence. We define two lists to index each vector in the set of features as follow:

$$\text{ON} = \{0,2,4\}\,, \quad \text{OFF} = \{1,3,5\}\,. \tag{3}$$

We compute the distance between each encoded feature vector on the true SETI cadence we have for every pair of ON targets (3×2 different combinations) and pair of OFF targets (another 3×2 different combinations). We compute their similarities, whereas for a pair of adjacent ON-OFF (3×3 different combinations) we compute the inverse. The indices i and j represent the ID of the observation and each goes from 0 to 5. The only condition is that $i \neq j$. We then average over $n = 21$ combinations to get the clustering losses:

$$\Lambda_S = \frac{1}{n}\Big(\sum_{i\in\text{ON}}\sum_{j\in\text{ON}} ||S_i - S_j||^2 + \sum_{i\in\text{OFF}}\sum_{j\in\text{OFF}} ||S_i - S_j||^2 + \sum_{i\in\text{ON}}\sum_{j\in\text{OFF}} \frac{1}{||S_i - S_j||^2}\Big)\,. \tag{4}$$

For the false SETI (RFI) cadence, since we inject a continuous signal across all six observations, we can assume that every pair of observations will look similar. Therefore, we compute the distance between each encoded feature vector for the RFI cadence by calculating the similarity between every pair and average over them to get the clustering losses:

$$\Lambda_R = \frac{1}{n}\Big(\sum_{i\in\text{ON}}\sum_{j\in\text{ON}} ||R_i - R_j||^2 + \sum_{i\in\text{OFF}}\sum_{j\in\text{OFF}} ||R_i - R_j||^2 + \sum_{i\in\text{ON}}\sum_{j\in\text{OFF}} ||R_i - R_j||^2\Big)\,. \tag{5}$$

A small loss value of Λ means larger distance between the two vectors. In other words, we can identify true ETI signals this way because the ON scans (A) will be separated furthest from the rest of the OFF scans (B, C, D), while at the same time keeping the As and the BCDs each close by.

By formulating the cost function in this manner, we help the neural network to make an "implicit" classification that groups relevant features together. Finally we include another hyperparameter weight α to tune the relative "importance" of the loss metric. We empirically set $\alpha = 10$ as it gave the best performing model. In summary, the loss function (L) of our model can be represented by:

$$L(\theta,\phi,\beta;x,z,s,r) = \overbrace{-\mathbb{E}_{q_\phi(z|x)}[\log p_\theta(x|z)]}^{\text{Reconstruction Loss}} + \overbrace{\beta(D_{KL}(q_\phi(z|x)||p(z)))}^{\text{KL Divergence}} + \alpha\big(\overbrace{\Lambda_S(s,s')}^{\text{SETI}} + \overbrace{\Lambda_D(r,r')}^{\text{RFI}}\big)\,. \tag{6}$$

3.3 Random Forest Decision Tree

After constructing the $\beta-$VAE we create a Random Forest classifier[21] to perform the classification. Random Forests use an ensemble of decision trees to build a classifier, with each tree acting as a vote of confidence. We employ a Random Forest in combination with our β-VAE network because we needed to turn the encoder into a classifier and a Random Forest model is the fastest to train and to execute. The greater the number of estimators (trees), the more robust a classifier is. We build a classic Random Forest using the SciPy library[8] with 1000 estimators and bootstrap it when constructing the individual trees. Empirically this gives the best run-time vs performance metric.

The Random Forest model requires separate training since it needs to be built specific to a neural network where the weights are held constant. This is achieved by taking the outputs encoded by the β-VAE and feeding the results into the Random Forest with synthetic labelled data. We create 12,000 false cadences and 12,000 true cadences (total 24,000) with a breakdown ratio of various labelled categories as described in Section 3.1 (see also, Table 1). The input data for the Random Forest model is a 6×8 feature matrix (6 observations by 8 latent features). Since Random Forest only works with input that is one dimensional, we unroll the matrix into a single 48×1 feature vector.

[8] https://www.scipy.org

4 Performance Evaluation

In order to evaluate the model performance, we need to generate various test benches of labelled data. These different evaluations are described in the following subsections, while Table 2 summarizes the properties of these test sets.

Test type	Source	Freq range (MHz)	N. Bgs	N. Tests	S/N range	N_{bin} S/N	|DR| (Hz/s)	N_{bin} DR
Other models	HIP 15638	1418.7−1475.1	4928	6000	20−80	30	0−8	n/a
Full-band & S/N	HIP 15638	1023.9−1926.3	78848	480000	20−80	30	0−8	n/a
Drift rate	HIP 15638	1023.9−1926.3	78848	320000	20−80	n/a	0.5−10	20
Latent Space	HIP 110750	1531.5−1587.9	4928	4928	n/a	n/a	n/a	n/a

Table 2. Summary of the simulated data parameters for the test benches. We list the test type for each test bench. The source name of the cadence and the frequency range used are shown. The corresponding number of unique background snippets (N. Bgs) is listed, from which we randomly draw the said number of test samples (N. Tests) to inject simulated signals. The S/N range for each case is shown as well as the number of S/N bins evenly distributed within the range — except when the S/N are randomly chosen from the range specified we denote the cell with "n/a". Similarly, we list the absolute value of drift rate (DR) and its number of bins. For the latent space test, no simulated signal is injected.

4.1 Comparing against other ML models

We test our algorithm against traditional ML approaches, including a support vector machine (SVM)[24], a Random Forest[21], a classic neural network[12] , and a 3-D convolutional neural network[12]. The SVM and the Random Forest are both classifiers and can only work with feature vectors as input, so we perform an additional step of applying principal component analysis (PCA) on the unrolled feature vectors. We implement these models using the python module SCIPY[9] with little or no further fine tuning.

We use a different target, HIP 15638, as the background (see Table 2). We pick a relatively RFI-free part of the band between 1418.7−1475.1 MHz and draw 4928 unique backgrounds within this range. This test bench consists of 6,000 simulated snippets, with the same ratio of labelled events as our training set described in Section 3, i.e. half False data, a quarter "True Single Shot" and a quarter "True Double Shot". The goal is to subject different ML models to this test bench and see how successful they are in recovering the correct signal classifications.

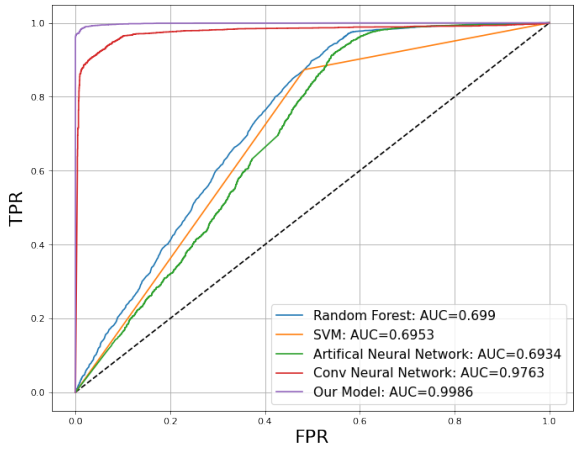


Figure 5. An ROC curve comparing the true positive rate (TPR) against the false positive rate (FPR) at various threshold settings for a number of ML models. The corresponding AUCs are listed in the legend. Models above the dashed line are considered better than random.

Fig. 5 shows a receiver operating characteristic (ROC) curve which evaluates the performance of each model at varying classification thresholds. All ML models perform well and are above the diagonal line. The Area Under the Curve (AUC)

[9]https://www.scipy.org/

provides an aggregate measure of performance summing over all classification thresholds. A perfect model would have an AUC of 1. For our ML model, we have an AUC of 0.9985, which is significantly higher than the other classical ML models. This provides evidence that simpler ML solutions perform less well than our β-VAE-based model. If we compare SVM and Random Forest, we also see that Random Forest performs better which validates our choice of using the Random Forest as the classifier.

4.2 Performance as a function of observing frequency, S/N and drift rate

In order to evaluate the model performance across the frequency band and across the S/N range, we create one test bench using an observation of HIP 15638 as the background (see Table 2). We split the full GBT data into 16 sub-bands and we step through the S/N range of 20 to 80 with a step size of 2, resulting in 30 S/N bins. We deliberately simulate signals brighter than the max S/N of 50 used in the initial training dataset. The idea is to see if our model has successfully generalized to high S/N signals that it was not previously trained on. For each frequency sub-band and each S/N bin, we simulate 1000 test samples. This amounts to a total of $16\times30\times1000$=48,000 simulated signals, again with the same ratio of labelled events and same ranges of widths and drift rates as our training set described in Section 3.

For the frequency test, we average over all S/N and compare across the 16 sub-bands. Since our test bench consists of balanced data, i.e., equal amount of False and True signals, we first study the performance at the standard probability threshold of 50%. From Fig. 6a, we can see that overall the accuracies are fairly high across the full band, with the majority of the sub-band having accuracy over 97.5%. In other words, False, Single True and Double True labelled data were correctly identified over 97.5% of the time. The drop near 1600 MHz corresponds to regions of the band that are heavily affected by RFI signals, as reported by[5] and as discussed later in Section 6.1.

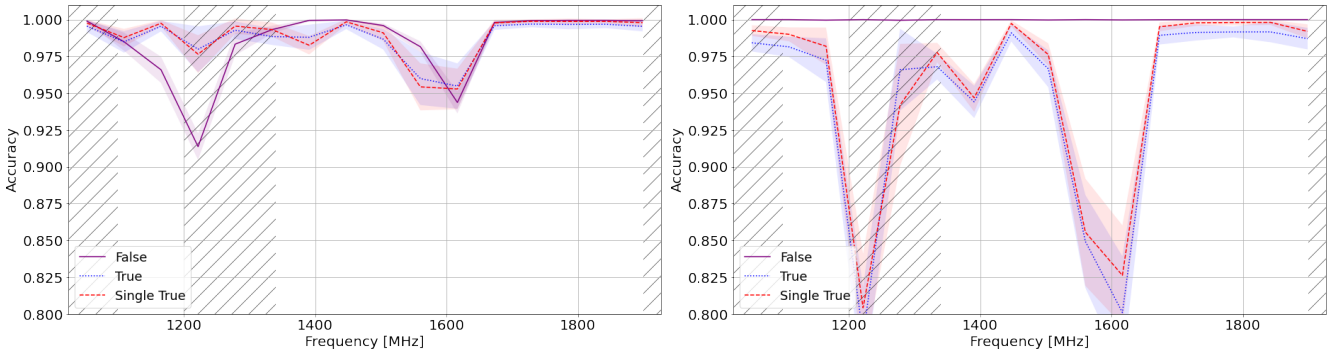


Figure 6. Performance evaluation across the full band for a classification threshold of (a) 50% and (b) 90% as a function of the frequency band. The regions of the band discarded due to instrumental artifacts are shaded in gray. The colored bands represent one standard deviation.

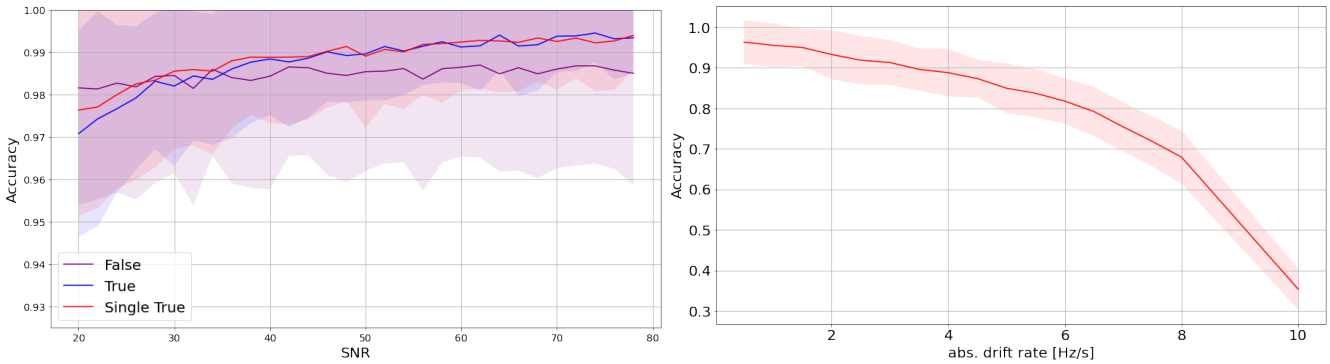


Figure 7. (left) Performance evaluation as a function of S/N for a threshold of 50%. We shade the regions that are affected by known GBT instrumental artifacts which are discarded from the analysis. The colored bands represent one standard deviation. (right) Performance evaluation on "Single True" signals as a function of drift rate for a threshold of 90%. The colored bands represent one standard deviation.

As mentioned earlier, high false positive rate has been a persistent bottleneck in SETI efforts. Even a false positive rate of a few percent can translate to tens of thousands of candidates, reducing the search efficiency. We can attempt to drive down

the false positive rate by selecting a higher classification threshold. Fig. 6b shows the model accuracy at a threshold of 90%, providing a $<1\%$ false positive rate with a maximum 1-σ uncertainty of 0.1% out of the 30×1000 samples per sub-band. Naturally, a higher threshold will also mean worse true positive detection, meaning that we have a slightly higher chance of missing genuine ETI signals in the data. However, we can see that the biggest drop in true positives happens in discarded regions of the band where prominent RFI is present. Barring those regions, we still have a true positive accuracy of above 95% in the rest of the band.

For the S/N performance evaluation, we average over all frequency sub-bands and compare across the 30 S/N bins. In the left panel of Fig. 7 we record a pretty consistent performance accuracy of $\gtrsim 95\%$ across S/N 20 to almost 80, with the largest uncertainty of $\pm2.7\%$ out of the 16×1000 samples per S/N bin.

For the drift rate performance evaluation, we simulated "Single True" signals between a drift rate range of 0.5 to 10 Hz/s with a step size of 0.5 Hz/s. At each drift rate, we loop through the 16 sub-bands each with 1000 samples of randomly selected S/N within the range of $20-80$. This sums to a total of 16,000 samples per drift rate. We can see from the right panel of Fig. 7 that our performance drops as the drift rate increases. At 8 Hz/s, we have roughly 70% accuracy in identifying "Single True" injected signals. This drops to about 35% at 10 Hz/s. This degradation can be explained by our unbalanced test samples discussed in Section 3.1, where fewer high drift rate signals were employed in the training of the ML model.

4.3 Latent Space Analysis

In order to understand and interpret how our ML algorithm makes classification decisions, we study the latent space of the decoder with the goal of finding out whether the network is able to disentangle each of the learned features. The first three axes appear to produce human interpretable results and so we further study their feature vectors and we perturb each axis in steps of 0.1. These variations are then fed back into the decoder and we visualize the resultant spectrograms in Fig. 8. Axis $z1$ looks to have encoded the feature of S/N as perturbing this axis creates a brighter signal. Axis $z2$ appears to represent the level of background noise and axis $z3$ corresponds to instrumental artifacts which are commonly seen as bright horizontal bands across part of the spectrum. We interpret this outcome as evidence that the network has learnt relevant features from the dataset, helping it make the right decisions during the classification stage.

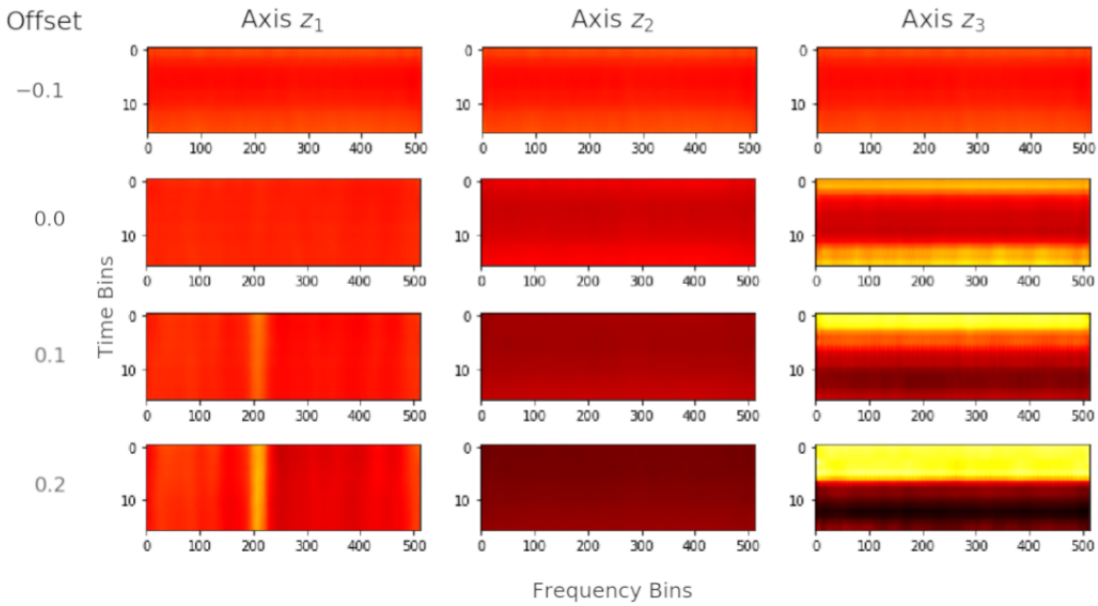


Figure 8. Spectrograms showing the effects of varying the first three axes of the latent space vectors. The perturbations are in steps of 0.1 and are indicated on the left hand side of the plots.

5 Search Pipeline

We develop a processing pipeline in order to systematically analyze the 120 TB of GBT 1.1–1.9 GHz data. Within each 4096-channel snippet, each search is completely independent and hence can be easily parallelized. We have access to three compute nodes at the Berkeley data centre for this analysis (see Table 3 for the hardware specifications), therefore we divide up the dataset into three equal sub-sets to run on the three machines independently as shown in Fig. 9. On each machine, we

launch five parallel instances of the processing that independently work on different cadences. The status of the processing on each machine is monitored and managed manually.

Initially, we divide up the full frequency band into 16 sub-bands. This allows us to load observations faster into memory compared to reading in and processing each 4096-channel snippet one by one, because of the switching cost during input/output (I/O) vs processing. The I/O was done using the python package BLIMPY[10] as described in[25]. We concatenate the six observations of a cadence before feeding it into the neural network, which is built using the Tensorflow API and is GPU-accelerated. Furthermore we distribute the training across multiple GPUs implementing a typical asynchronous mirrored approach. Because we mirror each model on different GPUs we can afford to execute the model on batch sizes of 10^4 snippets given the GPU memory constraints to further parallelize execution. Finally the features are fed into the Random Forest classifier which runs in parallel on the CPU.

The search pipeline, disregarding I/O, takes <5 s to search $1/16^{\text{th}}$ of the band. The I/O and the preprocessing take the most amount of compute time despite it being implemented to compute in parallel and with just-in-time compilers (JIT) such as NUMBA[11], taking $50-60$ s to run on the same $1/16^{\text{th}}$ band. The search time on a single 30 min cadence is 18.7 ± 1.6 min. The variation is mostly due to cache preloading for some data. However the runtime of the core of the search pipeline disregarding the I/O stays relatively constant from cadence to cadence.

We achieve further speed-up via a custom container orchestration using SINGULARITY[12] and the schematic is shown in Fig. 9. By running multiple instances of the same search on smaller subsystems on the same compute node, it scales faster in practice than running a single instance on all the resources[26]. This is because each search is completely independent from each other and hence does not require sharing memory or data. This is far more efficient than parallelizing individual operations such as I/O where data needs to then be recombined together.

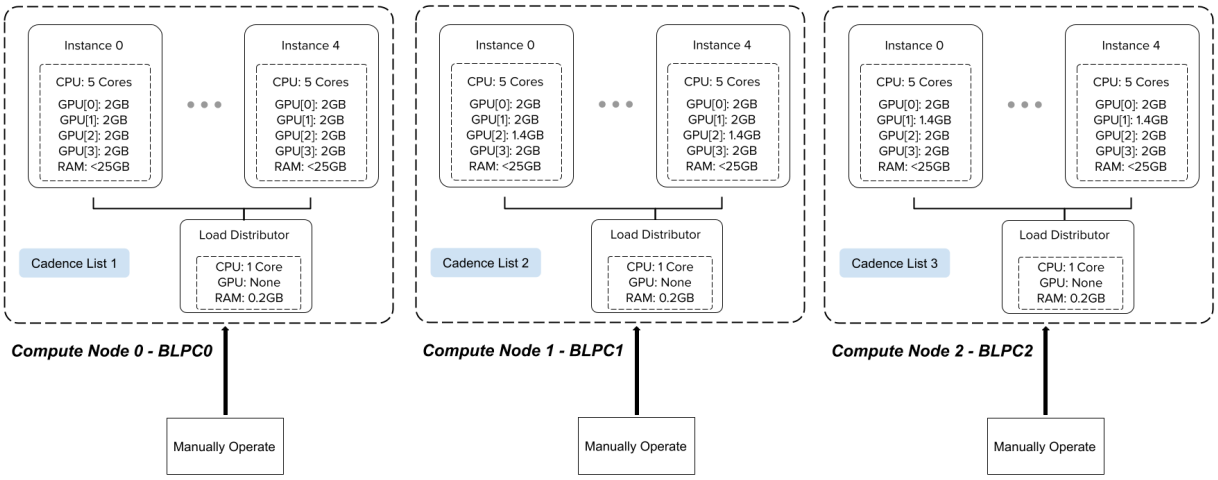


Figure 9. Figure depicts the orchestration of singularity containers and how the computing resources were distributed to perform individual searches.

	Compute Node 0	Compute Node 1	Compute Node 2
CPU	Intel Xeon E5-2630	Intel Xeon E5-2630	Intel Xeon Silver 4210
GPU[0]	NVIDIA TITAN X	NVIDIA TITAN X	NVIDIA TITAN X
GPU[1]	NVIDIA TITAN X	NVIDIA TITAN XP	NVIDIA GTX 1080
GPU[2]	NVIDIA TITAN X	NVIDIA GTX 1080	NVIDIA GTX 1080 Ti
GPU[3]	NVIDIA TITAN X	NVIDIA TITAN XP	NVIDIA GTX 1080 Ti
RAM	256 GB	256 GB	196 GB

Table 3. The hardware specifications for each compute node used in this analysis.

[10] https://github.com/UCBerkeleySETI/blimpy
[11] https://numba.pydata.org/
[12] https://sylabs.io/

6 Search Results

6.1 Further candidate filtering

Our search includes a total of 57 million unique snippets, excluding the regions of the band affected by instrumental effects as mentioned in Section 2. The additional overlap search means that we actually analyze close to 115 million snippet windows. From these, our ML model returns a total of 2,917,789 signals-of-interest. We visualize the distribution of these events as a function of observing frequency as shown in Fig. 10. It can be seen that certain observing frequencies contain a much higher number of events compared to the others — for example the region around 1600 MHz. This overlaps with known RFI at the GBT site specifically from persistent GPS signals. Those regions of the spectrum are heavily contaminated by RFI and it would be challenging to detect anything apart from RFI in those frequencies. Using the frequency histogram in the bottom left panel of Fig. 10 which has a histogram bin size of 4.97 MHz, we empirically determine a threshold to discard frequency bins with more than 35,000 events per bin, since this represents a conservative level where the main peaks of RFI clusters can be flagged. This equates to discarding 13 histogram bins, which is about 65 MHz of the entire band.

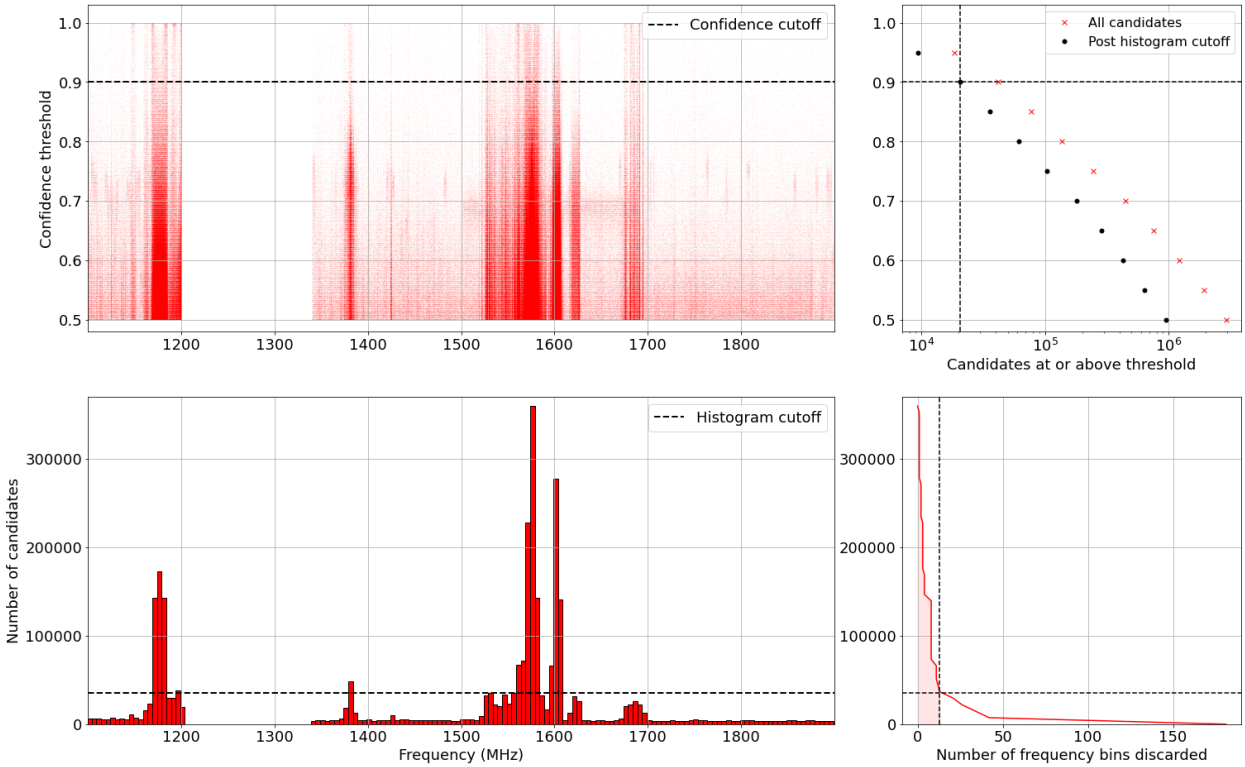


Figure 10. (top left) The distribution of signals-of-interest in terms of confidence threshold and observing frequency. (bottom left) A histogram of the number of signals-of-interest at each observing frequency, with histogram bin size of 4.97 MHz. (top right) The number of signals-of-interest at or above a given classification threshold. The red data points are without the frequency histogram-based filtering, whereas the black data points are after such filtering. (bottom right) The number of histogram bins being filtered out at progressively lower cutoff number of signals-of-interest.

The top right panel of Fig. 10 shows the number of signals-of-interest at a range of classification thresholds, which has a quadratically increasing trend towards low thresholds. Based on the discussion in Section 4.2, we further limit our signals-of-interest to those that have a classification threshold of over 90% in order to reduce the number of false positives. After applying these further filtering criteria, we are left with 20,515 signals-of-interest to assess.

6.2 Signal-of-interest visualization

In order to display the information for each signal-of-interest snippet and to visually assess them, we create diagnostic plots for each of the 20,515 events returned by the ML model using a Python script. Fig. 11 shows two examples of these diagrams, one for an event measured in the observation of HIP 54677 (described in Section 6.3 as MLc5, one of the top 8 signals-of-interest of our search) and one for an event in HIP 114456, which is ultimately rejected upon human assessment due to the non-uniform drift rate across the three ON observations.

The header contains information about both the whole cadence and the signal-of-interest snippet from that cadence returned by the ML model. The information on the cadence includes the catalog name of its target star (the ON observation), its celestial coordinates (right ascension and declination), the telescope used, the start time of the observation written in both modified Julian day (MJD) and in ISO 8601 format for date and time (ISOT), the time spent observing ON-target, the cadence type (ABABAB or ABACAD), as well as the the minimum and maximum frequency of this cadence recorded during the observation. Adding to this information are the number of events identified by the ML model for this specific cadence and the average number of events identified for all cadences. The information presented on the signal-of-interest includes the start and end frequencies of the snippet, the bandwidth of the snippet as well as the confidence rating according to the ML model. Each signal-of-interest is given a numerical identification composed of a number representing the cadence and a number representing the event snippet in that cadence. The header is juxtaposed with a QR code which encodes the ID of the signal-of-interest, the cadence target name, the start and end frequencies of the snippet as well as paths to the visualizer itself and to each of the six HDF5 files that make up the cadence. Paths correspond to locations on the Breakthrough Listen computers.

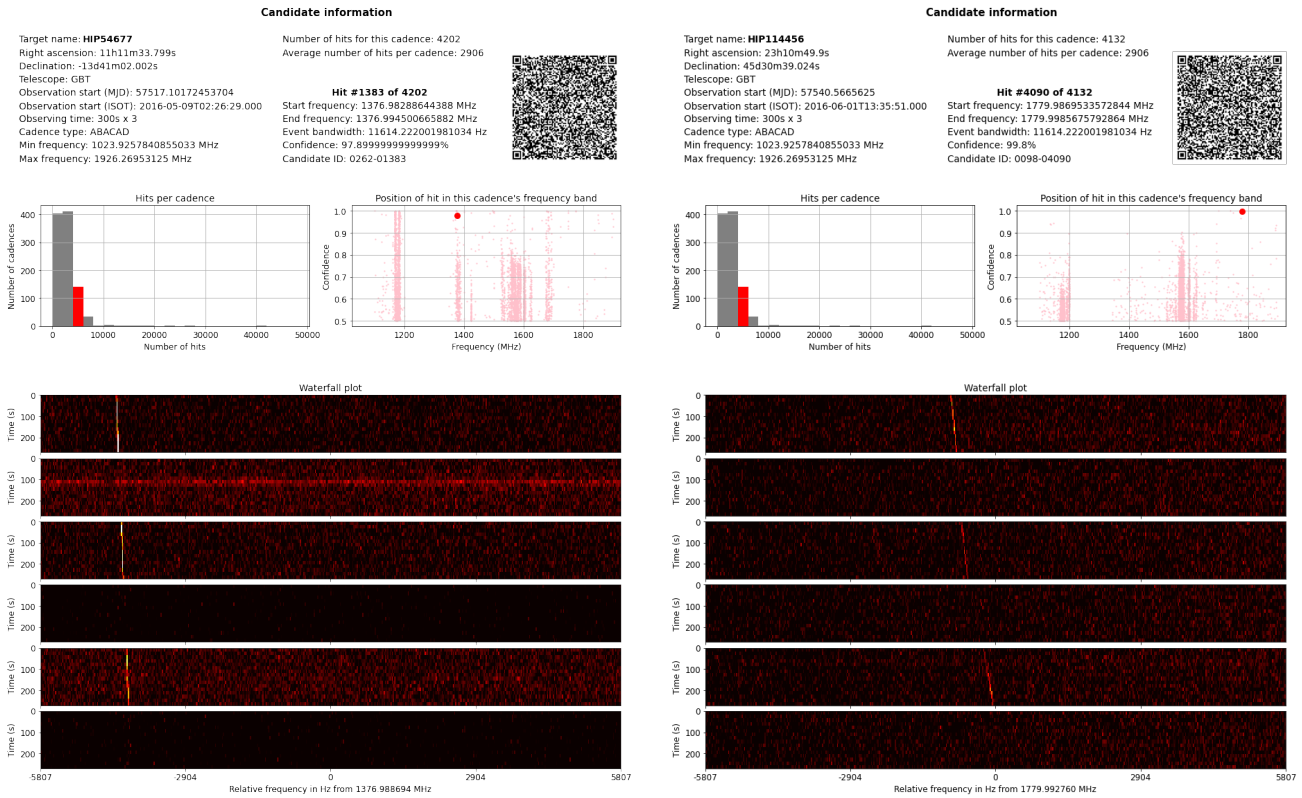


Figure 11. Two examples of the diagnostic plots. (Left) One of the top 8 signals-of-interest described in Section 6.3. (Right) An interesting event returned by the ML model that is rejected as SETI candidate upon human inspection because of the non-uniform drift rate across the three ON-scans.

Two analytical plots are included in the middle of the diagram. The plot on the left is a histogram which classifies the cadences by the number of events identified in them, with the bin of the cadence in question highlighted in red. This is used to give a sense whether the specific cadence contains an usually high amount of events (signals-of-interest) or not, as that might indicate an observation that is heavily contaminated by RFI and thus potentially less reliable. The plot on the right shows the positions of all events of a cadence in frequency space (i.e. where they fall in the bandwidth of the cadence) as well as the confidence ratings of the signals-of-interest. The signal-of-interest in question is represented by a large red dot in this plot. This plot helps to assess whether the specific signal-of-interest comes from a region of the frequency band that has a large number of events (higher chance that the signal-of-interest is also RFI) or if the region is relatively empty of events (higher chance that the signal-of-interest is genuinely special). Finally, the bottom half of the diagram is dedicated to the waterfall plot, a representation of frequency vs. time showing the signal-of-interest snippet across each of the six ON and OFF observations with lighter color signifying higher intensity. The waterfall plot is created using a modified version of the Python package

BLIMPY[13] and is downsampled by a factor of 8 in frequency, the same as the input dimension to the ML model. In other words, each frequency bin in the plot has a resolution about ~ 22.4 Hz.

All 20,515 diagrams generated are collated into an mp4 file to be viewed as a movie with four frames per second. The full video can be found at this weblink[14].

6.3 Promising Signals-of-interest

ID	Target	Center Freq. (MHz)	MJD	Confidence	Drift rate (Hz/s)	S/N
MLc1	HIP 13402	1188.545038	57541.68902	98.1	+1.11(25)	6.53
MLc2	HIP 118212	1347.868136	57752.78580	99.9	−0.44(7)	16.38
MLc3	HIP 62207	1351.631217	57543.08647	93.7	−0.05(10)	57.52
MLc4	HIP 54677	1372.987594	57517.08789	99.9	−0.11(3)	30.20
MLc5	HIP 54677	1376.994501	57517.09628	97.9	−0.11(2)	44.58
MLc6	HIP 56802	1435.946114	57522.13197	99.9	−0.18(4)	39.61
MLc7	HIP 13402	1487.487853	57544.51645	99.9	+0.10(2)	129.16
MLc8	HIP 62207	1724.978369	57543.10165	99.9	−0.126(10)	34.09

Table 4. The top 8 signals-of-interest identified by our ML model. We list the name of the ON-source target star, the frequency at the middle of the relevant ∼11 kHz snippet, the MJD of the detection, the topocentric drift rate, as well as the S/N of the signal.

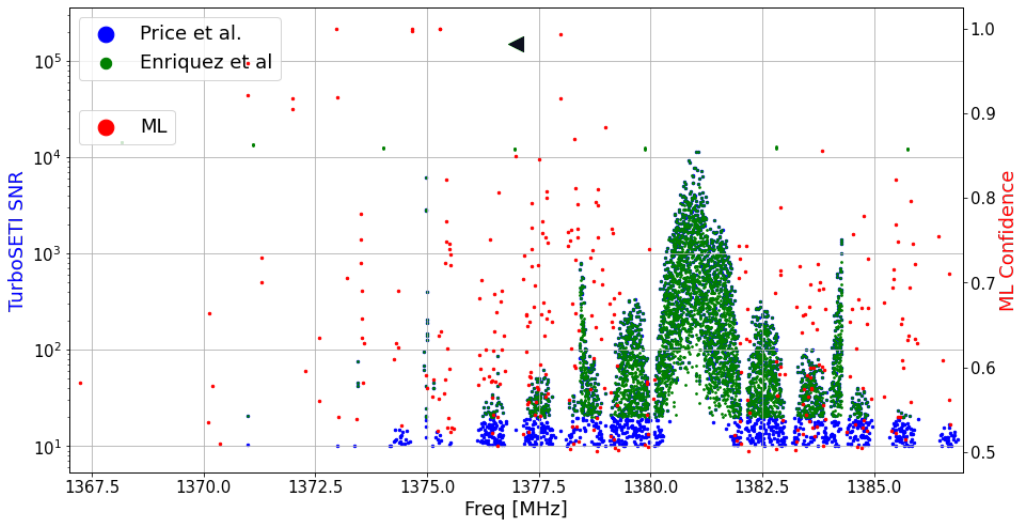


Figure 12. The S/N-confidence vs frequency distribution of the hits identified by TURBOSETI as reported by[7] (blue) and by[5] versus events detected by our ML model (red) in the cadence of target HIP 54677. The ML event that corresponds to MLc5 is marker by a black triangle symbol.

Upon a visual inspection, we identify 8 promising signals-of-interest that show narrow, drifted signals in the three ON-scans. Refer to Table 4 for their respective parameters. One of the signals-of-interest can be seen in the left panel of Fig. 11. We consider these 8 signals-of-interest potentially worthy of further analysis and follow up observations with other telescope facilities. These 8 signals-of-interest come from five different stars. HIP 13402 and 54677 both have spectral type K whereas HIP 62207 is a G star, HIP 56802 an F star and HIP 118212 is an M star. They are all within 30 to 90 ly from Earth. All of five targets were analyzed by[5] and by[7] but they did not detect anything similar. See Fig. 12 for a comparison of the hits

[13] https://github.com/UCBerkeleySETI/blimpy
[14] https://www.youtube.com/watch?v=iSdVfOwPVCI

(not yet grouped per cadence to be considered as events) reported by these two TURBOSETI searches versus the events found by our ML algorithm on target HIP 54677 in the frequency ranges where we have detected our ETI signal-of-interest MLc5. The region around the signal-of-interest is clearly empty of any other detections. We note that there is a chance that the signals-of-interest found near 1370−1380 MHz (e.g. MLc4 and MLc5) could be related to the GPS signal in that band, given that they also show similar drift rates. As can be seen in Fig. 12, the blue and green TurboSETI hits essentially outline the GPS signal in the observing band. In fact eight out of the 11 prime signals-of-interest presented in[5] are from that particular region of 1370−1380 MHz. However, given that the main goal of this work is to apply ML technique to identify signals with a specific pattern, we do not attempt to make a definite conclusion of whether these 8 signals are genuinely produced by ETI. Re-observations of these signals-of-interest are underway and we defer to a future study to discuss their nature.

We individually compute the drift rate and S/N of these signals-of-interest as they are not a by-product of our ML pipeline. We estimate the drift rate in each ON-scan separately using SETIGEN. By stepping through a range of trial drift rates, we find out the one that returns the highest S/N. The uncertainty of the drift rate is defined to be the deviation of drift rates among the three ON scans. A number of interesting events were rejected despite showing narrow band drifted signals as we find that the drift rates across the three ON-scans are not consistent (see, e.g. the right panel in Fig. 11). We note that a changing drift rate does not necessarily mean the variation is non-physical. However, here we will limit our search criteria to constant drift rate for simplicity.

7 Discussion

Comparing our search to TURBOSETI, we find that our ML mode is successful in returning fewer false positives and more convincing signals-of-interest. Part of this is due to the fact that our ML model is built to consider the cadence pattern as a whole, whereas TURBOSETI searches each observation separately to produce hits, which then require a secondary grouping stage to cluster hits that come from the same signal-of-interest. Once cadence clustering is included, the number of TURBOSETI events drops to about 16k as reported by[7] which is more comparable to our number. We also note that none of our top 8 signals-of-interest were identified by TURBOSETI. In fact, we compare the common targets searched by the two pipelines and find that for every cadence, on average 64% of the events (with a deviation of 15%) flagged by our ML were not found by[5] and on average 61% (with a deviation of 37%) were not found by[7]. It thus appears that the events identified by the two searches are quite distinct from each other (see also, Fig. 12). We are also able to cover a wider drift rate range. Given how we were able to complete this search in approximately a week, we could in principle repeat the search with even higher maximum drift rate by including more overlapping in the alternating snippets search or by reading in larger snippets at a time.

Our ML algorithm also has some disadvantages compared to TURBOSETI. One of them is that unlike TURBOSETI, we do not directly obtain drift rate and S/N as output of the pipeline. We also do not know precisely where the signal of interest lies in terms of observing frequency and only know within which 4096-channel snippet ($\sim$11.6 kHz width) it is. The assessment of these is currently handled at the human inspection stage which is not ideal. Another limitation of our ML implementation is that we have a non-uniform training set with fewer synthetic signals at higher drift rates (see Fig. 3). This means that our model is less well-trained at the high drift rate range, despite the architecture of the overlap search providing theoretical detectability out to $\pm$10 Hz/s. This has likely skewed our model and taught it that low drift rate events are more common, as evident by the drift rate evaluation test discussed in Section 4.2. In other words, to quantify the overall detectability of this ML model as a function of drift rate, we should take into account both the parameter coverage as provided by the overlap search (right panel of Fig. 1), as well as the sampling distribution of simulated data (right panel of Fig. 7). For example, we estimate that we have a $\sim$80% sensitivity towards a 6 Hz/s signal that was present in this dataset, a $\sim$60% sensitivity towards a 8 Hz/s signal and a $\sim$25% sensitivity towards a 10 Hz/s signal. Another oversight in our simulated data is that we did not take into account the time delay between individual observations in a cadence, during which the telescope slews from one source to another. As mentioned in Section 2, this gap is typically small and of the order of a minute. For signals with a low drift rate of say 1 Hz/s, this is not really an issue, since a minute of time delay would correspond to a shift of about 60 Hz which is only 2 or 3 frequency bins in the downsampled snippet used for the ML search. However, this issue becomes more prominent at high drift rates. For example, at the maximum detectable drift rate of 10 Hz/s, during the minute of gap the signal would have shifted by about 600 Hz, which is roughly 25 downsampled frequency bins. Indeed, among the top 8 signals-of-interest we identified in Section 6.3, all of them have somewhat low drift rates of < 0.5 Hz/s. However, it is also possible that highly drifted signals are rarer. For future attempts in similar ML searches, we suggest that the training set should be made with a uniform number of samples (balanced class) with respect to drift rate and that the time delay between observations should be taken into account in order to avoid any potential bias towards high drift rates.

In addition, there are a number of tunable parameters in our set up (e.g. the hyperparameter β, the number of convolutional layers, the choice of classification threshold). We have not exhaustively sampled these parameter spaces to determine the optimal value. We know that our model works and it has produced a number of promising ETI signals-of-interest, but it is possible that we could further improve the model with better fine tuning of these parameters. Another thing we could optimize

is the grouping of events detected. Small drift rate signal can potentially be seen by two adjacent overlapped snippets. Currently, each snippet is searched independently and we have not attempted to associate detections across them. Doing that would lower the number of events the pipeline returns and improve its efficiency. Judging from the signal-of-interest distribution as a function of frequency (see e.g., bottom left panel of Fig. 10 and Fig. 12), we notice that their signals-of-interest and hits are not randomly distributed but have a definitive pattern with respect to the observing frequency. We suggest that quantifying the density of narrow-band features detected in a particular region of spectrum using either ML or traditional algorithms could be a useful additional way of assessing the confidence of signals-of-interest. We also note that a small number of cadences were recorded with the ABABAB pattern, which means in principle we could swap the observations around and use the B scans as the ON-source scans to increase the number of unique stars searched. Although these B scans are pointing towards blank sky and may not necessarily correspond with nearby stars. Looking ahead, we hope to expand this ML technique to other Breakthrough Listen datasets to further increase the impact of ML on SETI. This includes other GBT and Parkes data, as well as the upcoming MeerKAT[27] and Very Large Array (VLA) interferometric SETI projects.

8 Conclusion

This work represents the most comprehensive ML-based technosignature search to date, and improves on previous work by finding signals of interest not detected before[5,7]. We generate synthetic, labelled data to train a β-VAE framework together with a Random Forest classifier. We observe some level of generalization in the trained model, as the latent space shows interpretable features and the model is able to correctly identify signals beyond the initial training parameter space. Overall we see a high degree of accuracy when the model is subjected to a different test bench, both across the frequency band as well as over a wide range of signal S/N. Our model also performs better in comparison to a number of classical ML models tested.

By analyzing 1004 cadences corresponding to approximately 115 million snippets recorded with the GBT 1.1–1.9 GHz receiver, our model returns 2.9 million events. Upon further filtering by discarding RFI affected frequency bins as well as by limiting the classification threshold to 90%, we further reduce the number of events to approximately 20,000. By visually inspecting the individual diagnostic plots, we discover 8 promising SETI signals-of-interest with narrow band, drifted signals showing the expected on-off pattern that were not previously found by TURBOSETI.

Acknowledgements

Breakthrough Listen is managed by the Breakthrough Initiatives, sponsored by the Breakthrough Prize Foundation. (http://www.breakthroughinitiatives.org) We are grateful to the staff of the Green Bank Observatory for their help with installation and commissioning of the Breakthrough Listen backend instrument and extensive support during Breakthrough Listen observations. P.M. was supported by the Laidlaw foundation which has funded this project as part of the undergraduate research and leadership funding initiative. We thank Yuhong Chen for his helpful discussion on the Machine Learning framework. The first author would also like to thank the kind support of Dr. Laurance Doyle and Dr. Sarah Marzen for their generous guidance and encouragement to him when he first began his research career.

Code Availability

The code is available for review here at (https://github.com/PetchMa/F-Engine_Search)

Competing Interests

The authors declare no competing interests.

Supplementary information

Diagnostic plots for the remaining seven signals-of-interest listed in Table 4.

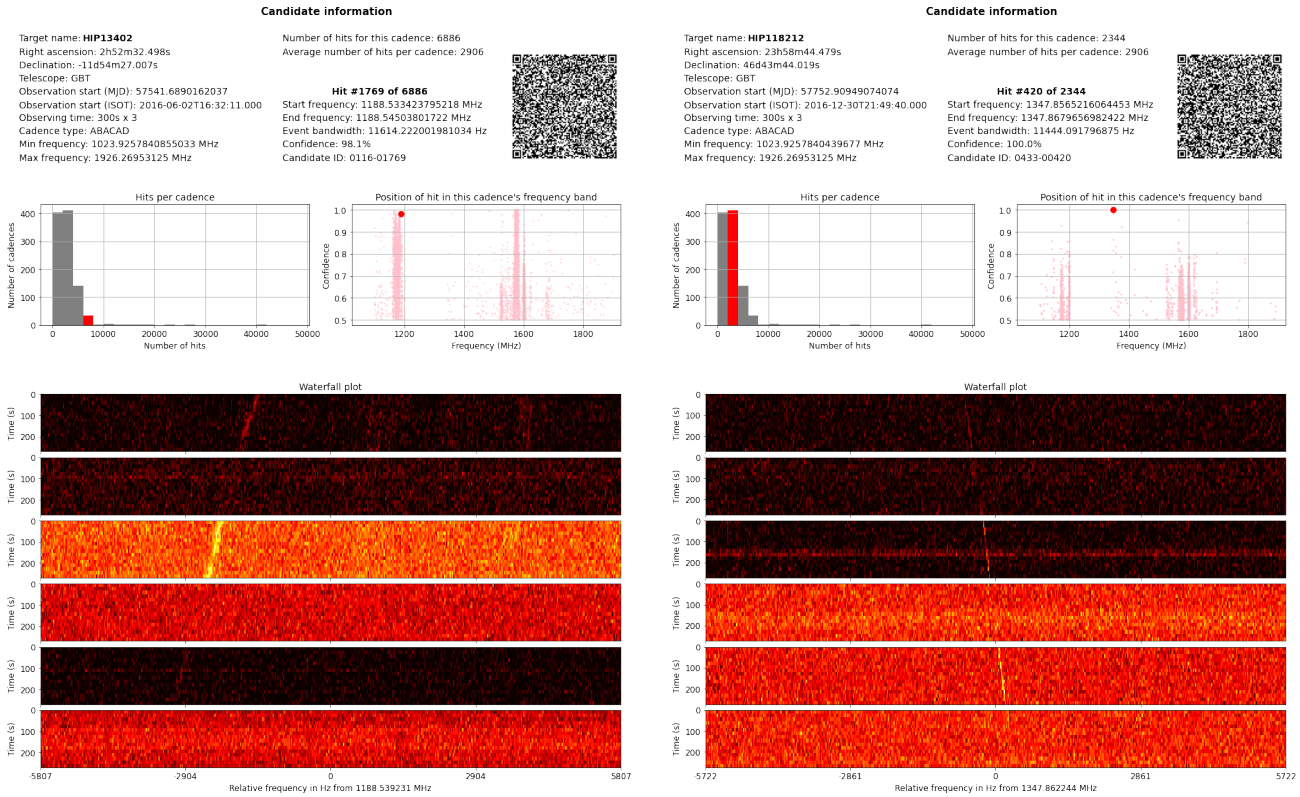


Figure 13. Diagnostic plots for (Left) MLc1 (Right) MLc2.

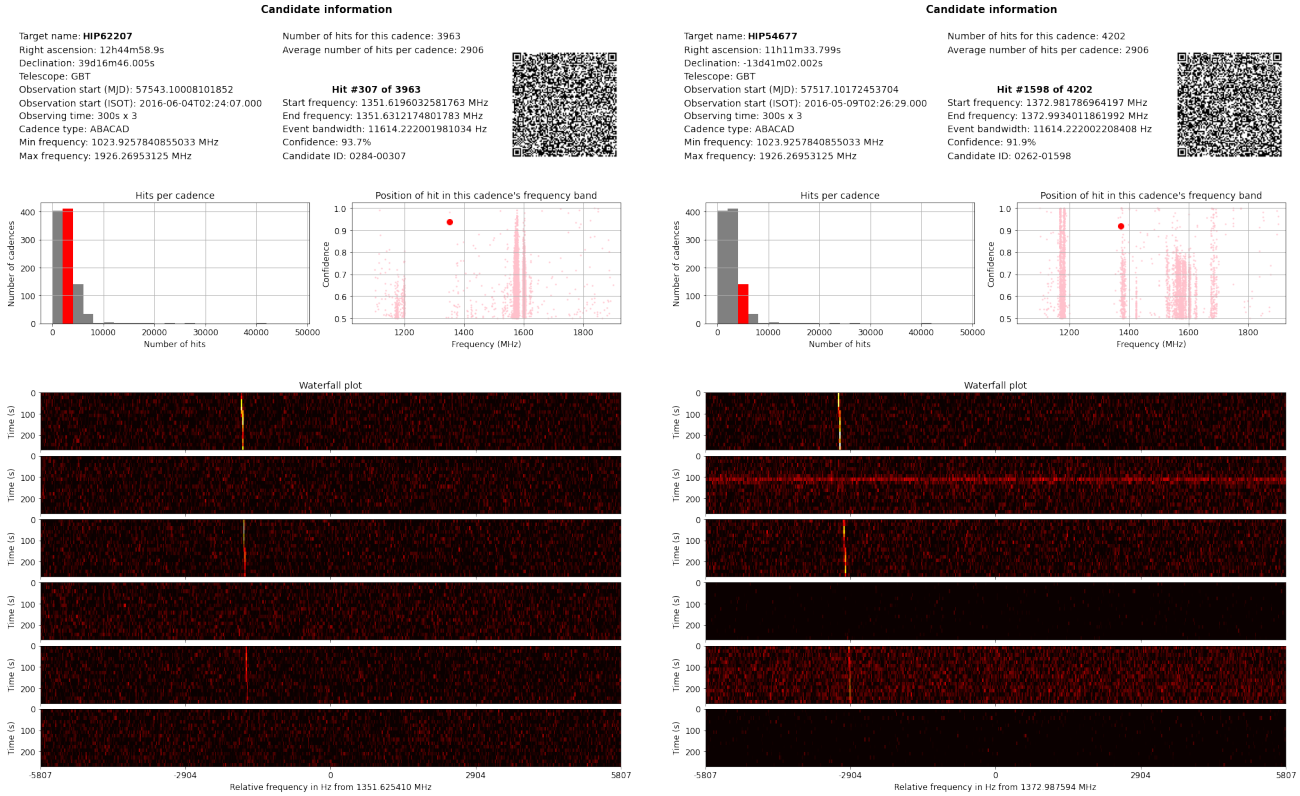


Figure 14. Diagnostic plots for (Left) MLc3 (Right) MLc4.

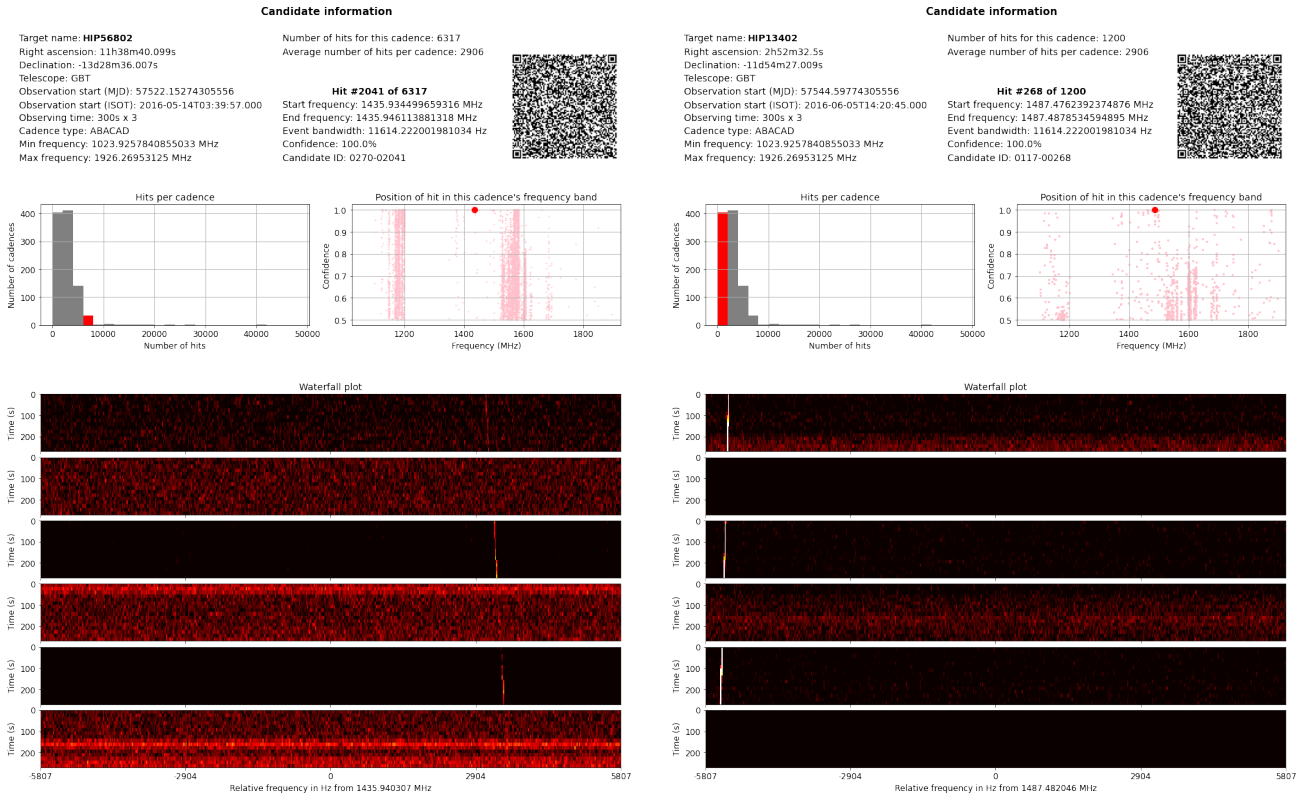


Figure 15. Diagnostic plots for (Left) MLc6 (Right) MLc7.

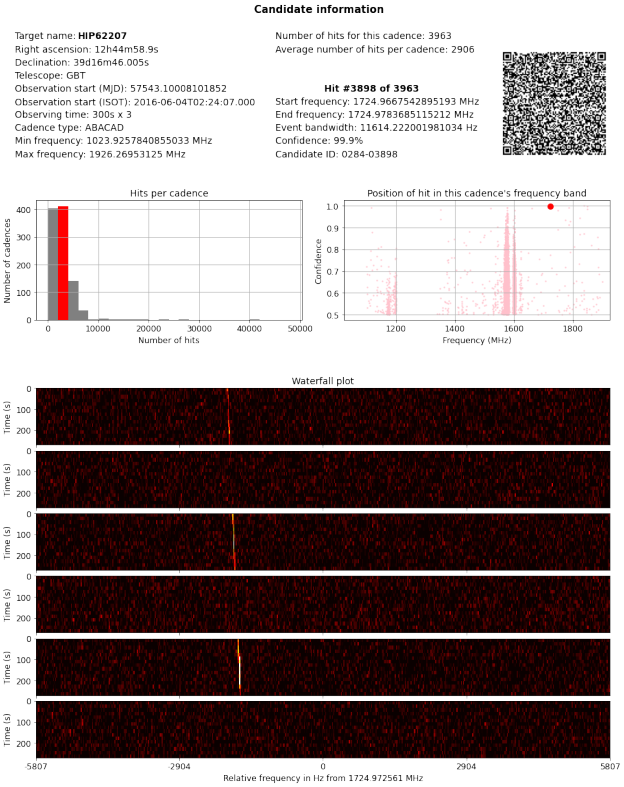


Figure 16. Diagnostic plot for MLc8.

References

1. Cocconi, G. & Morrison, P. Searching for Interstellar Communications. **184**, 844–846, DOI: 10.1038/184844a0 (1959).
2. Tarter, J. The Search for Extraterrestrial Intelligence (SETI). **39**, 511–548, DOI: 10.1146/annurev.astro.39.1.511 (2001).
3. Drake, F. D. Project Ozma. *Physics Today* **14**, 40, DOI: 10.1063/1.3057500 (1961).
4. Horowitz, P. & Sagan, C. Five Years of Project META: an All-Sky Narrow-Band Radio Search for Extraterrestrial Signals. **415**, 218, DOI: 10.1086/173157 (1993).
5. Enriquez, J. E. *et al.* The Breakthrough Listen Search for Intelligent Life: 1.1–1.9 GHz Observations of 692 Nearby Stars. *ApJ* **849**, 104, DOI: 10.3847/1538-4357/aa8d1b (2017). Publisher: American Astronomical Society.
6. Price, D. C. *et al.* The Breakthrough Listen search for intelligent life: Wide-bandwidth digital instrumentation for the CSIRO Parkes 64-m telescope. *Publications Astronomical Society Australia* **35**, DOI: 10.1017/pasa.2018.36 (2018). Publisher: Cambridge University Press.
7. Price, D. C. *et al.* The Breakthrough Listen Search for Intelligent Life: Observations of 1327 Nearby Stars Over 1.10-3.45 GHz. **159**, 86, DOI: 10.3847/1538-3881/ab65f1 (2020). 1906.07750.
8. Price, D. C. *et al.* Expanded Capability of the Breakthrough Listen Parkes Data Recorder for Observations with the UWL Receiver. *Research Notes American Astronomical Society* **5**, 114, DOI: 10.3847/2515-5172/ac00c1 (2021). ADS Bibcode: 2021RNAAS...5..114P.
9. Lebofsky, M. *et al.* The Breakthrough Listen Search for Intelligent Life: Public Data, Formats, Reduction, and Archiving. *PASP* **131**, 124505, DOI: 10.1088/1538-3873/ab3e82 (2019). Publisher: IOP Publishing.
10. Enriquez, E. & Price, D. turboSETI: Python-based SETI search algorithm (2019). 1906.006.
11. Siemion, A. P. V. *et al.* A 1.1-1.9 GHz SETI Survey of the Kepler Field. I. A Search for Narrow-band Emission from Select Targets. **767**, 94, DOI: 10.1088/0004-637X/767/1/94 (2013). 1302.0845.
12. LeCun, Y., Haffner, P., Bottou, L. & Bengio, Y. Object Recognition with Gradient-Based Learning. In Forsyth, D. A., Mundy, J. L., di Gesú, V. & Cipolla, R. (eds.) *Shape, Contour and Grouping in Computer Vision*, Lecture Notes in Computer Science, 319–345, DOI: 10.1007/3-540-46805-6_19 (Springer, Berlin, Heidelberg, 1999).
13. Khalifa, N. E., Taha, M., Hassanien, A. E. & Selim, I. Deep galaxy: Classification of galaxies based on deep convolutional neural networks. (2017).
14. Harp, G. R. *et al.* Machine Vision and Deep Learning for Classification of Radio SETI Signals. *arXiv e-prints* arXiv:1902.02426 (2019). 1902.02426.
15. Pinchuk, P. & Margot, J.-L. A Machine-Learning-Based Direction-of-Origin Filter for the Identification of Radio Frequency Interference in the Search for Technosignatures. *arXiv e-prints* arXiv:2108.00559 (2021). 2108.00559.
16. Zhang, Y. G., Won, K. H., Son, S. W., Siemion, A. & Croft, S. Self-supervised Anomaly Detection for Narrowband SETI. *arXiv:1901.04636 [astro-ph]* (2019). ArXiv: 1901.04636.
17. Brzycki, B. *et al.* Narrow-band Signal Localization for SETI on Noisy Synthetic Spectrogram Data. **132**, 114501, DOI: 10.1088/1538-3873/abaaf7 (2020). 2006.04362.
18. Srivastava, N., Mansimov, E. & Salakhutdinov, R. Unsupervised Learning of Video Representations using LSTMs. *arXiv e-prints* arXiv:1502.04681 (2015). 1502.04681.
19. Kingma, D. P. & Welling, M. Auto-Encoding Variational Bayes. *arXiv:1312.6114 [cs, stat]* (2014). ArXiv: 1312.6114.
20. Higgins, I. *et al.* -VAE: LEARNING BASIC VISUAL CONCEPTS WITH A CONSTRAINED VARIATIONAL FRAMEWORK. 22 (2017).
21. Breiman, L. Random Forests. *Machine Learning* **45**, 5–32, DOI: 10.1023/A:1010933404324 (2001).
22. Perryman, M. A. C. *et al.* The Hipparcos Catalogue. **500**, 501–504 (1997).
23. Snoek, J., Larochelle, H. & Adams, R. P. Practical bayesian optimization of machine learning algorithms. *Advances neural information processing systems* **25** (2012).
24. Cristianini, N. & Ricci, E. Support Vector Machines. In Kao, M.-Y. (ed.) *Encyclopedia of Algorithms*, 928–932, DOI: 10.1007/978-0-387-30162-4_415 (Springer US, Boston, MA, 2008).
25. Price, D., Enriquez, J., Chen, Y. & Siebert, M. Blimpy: Breakthrough Listen I/O Methods for Python. *The Journal Open Source Software* **4**, 1554, DOI: 10.21105/joss.01554 (2019).

26. Sochat, V. Singularity Compose: Orchestration for Singularity Instances. *JOSS* **4**, 1578, DOI: 10.21105/joss.01578 (2019).

27. Czech, D. *et al.* The Breakthrough Listen Search for Intelligent Life: MeerKAT Target Selection. **133**, 064502, DOI: 10.1088/1538-3873/abf329 (2021). 2103.16250.