Machine learning astrophysics from 21 cm lightcones: impact of network architectures and signal contamination

David Prelogović, Andrei Mesinger, Steven Murray, Giuseppe Fiameni, Nicolas Gillet

Monthly Notices of the ROYAL ASTRONOMICAL SOCIETY

MNRAS **509,** 3852–3867 (2022) https://doi.org/10.1093/mnras/stab3215
Advance Access publication 2021 November 12

Machine learning astrophysics from 21 cm lightcones: impact of network architectures and signal contamination

David Prelogović [1]★ Andrei Mesinger [1] Steven Murray [2] Giuseppe Fiameni[3] and Nicolas Gillet [4]

[1]*Scuola Normale Superiore, Piazza dei Cavalieri 7, I-56126 Pisa, Italy*
[2]*School of Earth and Space Exploration, Arizona State University, 781 Terrace Mall, Tempe, AZ, 85287, USA*
[3]*NVIDIA AI Technology Center Italy, Via Melchiorre Gioia 8, I-820124 Milan, Italy*
[4]*Observatoire Astronomique de Strasbourg, Université de Strasbourg, 11 rue de l'Université, F-67000 Strasbourg, France*

Accepted 2021 November 2. Received 2021 October 1; in original form 2021 July 1

ABSTRACT

Imaging the cosmic 21 cm signal will map out the first billion years of our Universe. The resulting 3D lightcone (LC) will encode the properties of the unseen first galaxies and physical cosmology. Here, we build on previous work using neural networks (NNs) to infer astrophysical parameters directly from 21 cm LC images. We introduce recurrent neural networks (RNNs), capable of efficiently characterizing the evolution along the redshift axis of 21 cm LC images. Using a large database of simulated cosmic 21 cm LCs, we compare the relative performance in parameter estimation of different network architectures. These including two types of RNNs, which differ in their complexity, as well as a more traditional convolutional neural network (CNN). For the ideal case of no instrumental effects, our simplest and easiest to train RNN performs the best, with a mean squared parameter estimation error (MSE) that is lower by a factor of $\gtrsim$2 compared with the other architectures studied here, and a factor of $\gtrsim$8 lower than the previously-studied CNN. We also corrupt the cosmic signal by adding noise expected from a 1000 h integration with the Square Kilometre Array, as well as excising a foreground-contaminated 'horizon wedge'. Parameter prediction errors increase when the NNs are trained on these contaminated LC images, though recovery is still good even in the most pessimistic case (with $R^2 \gtrsim 0.5-0.95$). However, we find no notable differences in performance between network architectures on the contaminated images. We argue this is due to the size of our data set, highlighting the need for larger data sets and/or better data augmentation in order to maximize the potential of NNs in 21 cm parameter estimation.

Key words: cosmology: theory – cosmology: observations – dark ages, reionization, first stars – early Universe – galaxies: high-redshift – methods: data analysis.

1 INTRODUCTION

Tomography using the hyperfine transition of HI is set to revolutionize our studies of the Cosmic Dawn (CD) and subsequent Epoch of Reionization (EoR). Although current radio telescopes are aiming for a statistical detection, the upcoming Square Kilometre Array (SKA)[1] will eventually provide a 3D image of the first billion years of our Universe. This image encodes the properties of the first generations of galaxies, whose UV and X-ray radiation fields imprint multiscale patterns in the 21 cm signal (see a recent review in Mesinger 2019).

How can we best interpret these patterns to learn about astrophysics and cosmology? Bayesian inference has recently become established in the field, either by directly forward-modelling the 21 cm lightcone (e.g. Greig & Mesinger 2017, 2018; Park et al. 2019; Greig et al. 2021) or through the use of emulators of the 21 cm power spectrum (e.g. Kern et al. 2017; Schmit & Pritchard 2018; Jennings et al. 2019; Ghara et al. 2020; Mondal et al. 2020). However, the question of most constraining summary statistic to use when comparing theory to data is as-yet unsettled. The huge data volumes and the fact that we do not know the initial seed of the Universe necessitate some form of compression of 21 cm images. However, the fact that the images are notably non-Gaussian (e.g. see fig. 1 in Mellema et al. 2015) means that there is additional information contained in the phases of wave-modes that is ignored in the commonly-used power-spectrum statistic. Indeed several studies have shown that non-Gaussian and morphological statistics contain complimentary information, and can improve parameter inference when combined with the power spectrum (e.g. Gazagnes, Koopmans & Wilkinson 2021; Watkinson, Greig & Mesinger 2021).

So what is the 'optimal' statistic for constraining astrophysics and cosmology from 21 cm images? Several candidate statistics (e.g. Shimabukuro et al. 2015, 2017; Majumdar et al. 2018; Giri et al. 2018a, 2019; Gorce & Pritchard 2019; Watkinson et al. 2019, 2021; Gazagnes et al. 2021) have been investigated in the literature, but these investigations have not performed a systematic treatment of the 'optimality' of the various statistics. Indeed without a strong a priori physical motivation, an optimal statistics is unlikely to be found.

An alternative approach is provided by deep learning techniques, specifically neural networks (NNs). By minimizing a loss function (the prediction error), NN can adaptively find a summary statistic

★ E-mail: david.prelogovic@sns.it
[1]https://www.skatelescope.org

that provides the most accurate parameter recovery. This comes with the downside that the resulting compression is difficult to interpret physically, and thus the performance strongly depends on having a large, representative data base for training.

NNs are rapidly becoming popular in the 21 cm community, recovering the underlying astrophysics and cosmology from 21 cm images. Gillet et al. (2019) introduced NN to the field, using 2D Convolutional Neural Networks (CNNs; LeCun et al. 1989) on idealized lightcone images. Although they did not consider instrumental effects, the data base of Gillet et al. (2019) spanned the largest variation in the cosmic signal, as they considered both EoR and X-ray heating parameters. La Plante & Ntampaka (2019) used a similar 2D CNN architecture to predict EoR timing and duration after removing foreground-contaminated modes. Kwon, Hong & Park (2020) predicted the mean neutral fraction during the EoR from 2D slices contaminated with Gaussian SKA noise. Better realism of SKA1-Low telescope effects was then used with 2D CNNs to constrain cosmology and astrophysics (Hassan, Andrianomena & Doughty 2020) and the neutral fraction (Mangena, Hassan & Santos 2020). Prediction uncertainties were introduced by Hortúa, Malago & Volpi (2020), using Bayesian Neural Networks to constrain cosmological and reionization parameters, however without instrumental effects. Zhao et al. (2021), on the other hand, used likelihood free Bayesian inference to retrieve posteriors of two idealized reionization parameters. In addition to these regression studies, CNN-based UNets were used to identify ionized regions during the EoR and remove foreground contamination from mock 21 cm images (e.g. Makinen et al. 2021; Bianco et al. 2021; Gagnon-Hartman et al. 2021).

However, the importance of the network architecture is often underappreciated. The above studies use CNNs to process 21 cm images, since they are capable of picking up correlations over a range of scales. Indeed, CNNs are very popular in computer vision, such as object detection, image semantic segmentation, and facial recognition. However, EoR/CD radio images have rather unusual properties: they are three-dimensional and *anisotropic*. Foreground and instrument systematics leave different imprints in the sky-plane and the frequency axis. More fundamentally, the lightcones of the cosmic signal have redshift evolution along the frequency axis, introducing strong correlations between neighbouring frequency bins.

In this work, we introduce long short-term memory (LSTM; Hochreiter & Schmidhuber 1997) recurrent neural networks (RNNs) for parameter estimation from 21 cm images. RNNs efficiently encode local correlations in sequential data that would otherwise require deeper (and thus more complex) network layers (for details, see Appendix A). As such, RNNs have become very popular in applications with temporal evolution between images/frames (e.g. language, audio, video; see the review in Schmidt 2019). Here, we exploit the directional differences in 21 cm lightcones by coupling 'traditional' 2D CNNs for the sky-plane with recurrent layers for the frequency dimension that encodes space–time evolution. We compare the performance of different architectures, including RNNs and a variant of 3D CNNs, in parameter estimation from 21 cm lightcones with varying degrees of signal contamination: (i) the cosmic signal only; (ii) signal + telescope noise; (iii) signal + telescope noise + foreground wedge excision.

This paper is organized as follows. In Section 2, we discuss how 21 cm data sets are generated. The motivation behind different network architectures is discussed in Section 3. The training procedure and performance are presented in Section 4. In Section 5, we quantify the parameter recovery for different architectures and levels of signal contamination. Finally, concluding remarks are presented in Section 6. Additional details on LSTM structure and architectures used can be found in Appendices Sections A and B, respectively. All quantities are quoted in co-moving units assuming ΛCDM cosmology: $(\Omega_\Lambda, \Omega_M, \Omega_b, n, \sigma_8, H_0) = (0.69, 0.31, 0.048, 0.97, 0.81, 68\ \mathrm{km\ s^{-1}Mpc^{-1}})$, consistent with the results from Planck Collaboration et al. (2020).

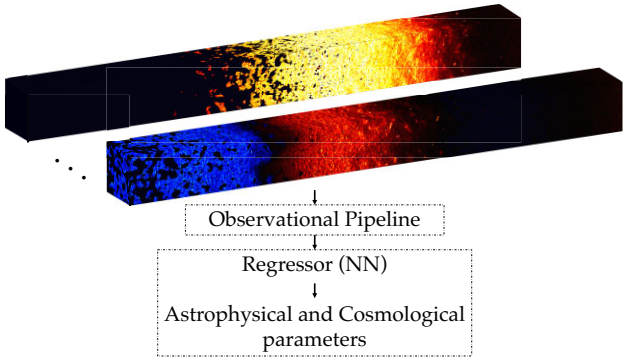


Figure 1. Generic pipeline for parameter estimation with NNs. A data base of cosmic signal lightcones is processed through an observational pipeline, including noise and systematics. The resulting data base of 21 cm images is fed into a regressor (neural network), trained to predict a 'best guess' for astrophysical and cosmological parameters.

2 DATA BASES OF MOCK 21 CM IMAGES

Fig. 1 shows the sketch of our procedure for creating a data base of mock 21 cm lightcones. We sample astrophysical parameters and cosmological initial seeds to compute 3D lightcones of the cosmological 21 cm signal (Section 2.1). Each lightcone is then passed through an observational pipeline (Section 2.2), including the following steps: (i) mean removal; (ii) addition of thermal noise; and (iii) removal of the foreground wedge. As a result, we generate three databases, corresponding to steps (i), (i)+(ii), and (i)+(ii)+(iii). These are then used to train an NN to predict astrophysical parameters. Below we describe each of these steps in turn.

2.1 Cosmological 21 cm signal

The cosmological 21 cm signal is defined as the brightness temperature offset with respect to the CMB, $\delta T_b \equiv T_b - T_\gamma$. It can be expressed as (e.g. Furlanetto, Oh & Briggs 2006):

$$\begin{aligned}\delta T_b(\nu) &= \frac{T_S - T_\gamma}{1+z}\left(1 - e^{-\tau_{\nu_0}}\right)\\ &\approx 27\, x_{HI}\,(1+\delta_{nl})\left(\frac{H}{\mathrm{d}v_r/\mathrm{d}r + H}\right)\left(1 - \frac{T_\gamma}{T_S}\right)\\ &\times \left(\frac{1+z}{10}\,\frac{0.15}{\Omega_M h^2}\right)^{1/2}\left(\frac{\Omega_b h^2}{0.023}\right)\,[\mathrm{mK}]\,.\end{aligned} \tag{1}$$

Here, T_S is the gas spin temperature, defining the 21 cm level population as $n_1/n_0 = 3\exp(-0.068\mathrm{K}/T_S)$, τ_{ν_0} is the optical depth at the 21 cm frequency ν_0, $1 + \delta_{nl}(\boldsymbol{x}, z) \equiv \rho/\bar{\rho}$ is the overdensity, $H(z)$ the Hubble parameter and $\mathrm{d}v_r/\mathrm{d}r$ the comoving gradient of the line of sight component of the comoving velocity. All quantities are evaluated at redshift $z = \nu_0/\nu - 1$.

We use the data base of cosmic 21 cm signals from Gillet et al. (2019), consisting of 10 000 lightcone images with a box

size of 300 Mpc and spatial resolution of 1.5 Mpc, simulated using the public, semi-numerical code `21cmFASTv2`[2] (Mesinger & Furlanetto 2007; Mesinger, Furlanetto & Cen 2011). For a given set of astrophysical parameters and choice of cosmological initial seed, the code generates a 3D lightcone of $\delta T_b(x, y, \nu)$, where the first two dimensions correspond to on-sky coordinates, while the third corresponds to the frequency (redshift) dimension (see Fig. 1). This calculation involves generating the initial density and velocity fields, which are then evolved using Lagrangian Perturbation Theory (e.g. Zel'Dovich 1970). The spatially dependent source field is computed from the evolved density using conditional halo mass functions (e.g. Barkana & Loeb 2004). For a given set of galaxy parameters, the inhomogenous reionization field is obtained by comparing the cumulative number of ionizing photons to the number of recombinations (e.g. Sobacchi & Mesinger 2014) in regions of decreasing radii (e.g. Furlanetto, Zaldarriaga & Hernquist 2004). Photons with long mean free paths, such as the soft UV and X-rays, are instead tracked by integrating the local emissivity back along the lightcone, for each simulation cell. Soft UV photons are attenuated using 'picket-fence' IGM absorption by the Lyman lines, while X-ray photons are attenuated by the partially ionized hydrogen and helium in the neutral component of the two-phased IGM. For more details on these calculations, interested readers are encouraged to see (Mesinger & Furlanetto 2007; Mesinger et al. 2011).

The data base of Gillet et al. (2019) varies four astrophysical parameters (in addition to co-varying the random seed), chosen to have both a clear physical meaning as well as driving the largest expected variation in the signal:

(i) $\zeta \in [10, 250]$, the UV ionizing efficiency of galaxies. It mainly controls the timing of the EoR, where higher values ionize the Universe earlier. It can be expanded as

$$\zeta = 30 \frac{f_{\rm esc}}{0.1} \frac{f_*}{0.05} \frac{N_{\gamma/{\rm b}}}{4000} \frac{1.5}{1+n_{\rm rec}}, \tag{2}$$

where the RHS corresponds to the following population-averaged quantities: $f_{\rm esc}$ is the fraction of ionizing photons escaping the host galaxy into the IGM, f_* is the fraction of galactic gas in stars, $N_{\gamma/{\rm b}}$ number of photons per baryon produced in stars, and $n_{\rm rec}$ is the average number of times a hydrogen atom recombines during the EoR. We use a constant ζ in order to allow direct comparison with previous work, though we note that `21cmFASTv2` allows ζ to scale with halo mass.

(ii) $T_{\rm vir} \in [10^4, 10^6]$ K, the minimum virial temperature of haloes hosting efficiently star-forming galaxies. Smaller haloes have suppressed star formation due to inefficient cooling and/or feedback. $T_{\rm vir}$ controls timing of all of astrophysical epochs. Moreover, it also impacts the characteristic scales of the heated/ionized regions, since it parametrizes the typical bias of the relevant galaxy population.

(iii) $L_{\rm X<2keV}/{\rm SFR} \in [10^{38}, 10^{42}]\,{\rm erg\,s^{-1}\,M_\odot^{-1}\,yr}$, the soft-band (with energies <2keV) X-ray luminosity per unit star formation rate (SFR). X-rays are responsible for heating the neutral IGM pre-reionization, and the resulting temperature fluctuations could drive the largest variance in the 21 cm signal (e.g. Mesinger, Ewall-Wice & Hewitt 2014). $L_{\rm X<2keV}/{\rm SFR}$ controls the timing of the epoch of heating (EoH) and correspondingly the overlap between the other astrophysical epochs of EoR and Lyman alpha coupling. Because of the cross terms in these fields, the level of overlap impacts the variance and dynamic range of the 21 cm signal.

(iv) $E_0 \in [0.1, 1.5]$ keV, the minimum energy of X-ray photons escaping the host galaxy. Photons of lower energies are absorbed in the ISM of the host galaxies, and thus do not contribute to the EoH. Because E_0 determines the typical mean free path of X-ray photons, it influences the timing and morphology of EoH, i.e. how homogeneously the IGM is heated.

Data sets are generated by sampling from flat distributions over the quoted ranges. $T_{\rm vir}$ and $L_{\rm X<2keV}/{\rm SFR}$ are sampled in log space, while ζ and E_0 are sampled in linear space.

We stress that it is important to model all of the astrophysical epochs for robust inference from 21 cm images. In particular, it is common for machine learning studies to focus only on the EoR, under the assumption that $T_{\rm S} \gg T_\gamma$ is valid during the EoR [c.f. equation (1); though see Gillet et al. 2019; Hortúa et al. 2020 who self-consistently compute the temperature]. However, recent studies have pointed out that $T_{\rm S} \gg T_\gamma$ is unlikely to be true during most of reionization. The observed galaxy UV luminosity functions (LF) imply a decreasing star formation efficiency with halo mass (e.g. Mirocha, Furlanetto & Sun 2017; Park et al. 2020), that in turn suggests a later epoch of heating than initial estimates. Although too simplistic to fully describe high-z galaxies, the parametrization we use in this proof-of-concept study does contain 'tuning knobs' for the relative timings and morphologies of all astrophysical epochs probed by the cosmic 21 cm signal.

2.2 Simple observational pipeline

Starting from the data base of cosmic 21 cm lightcones described in the previous section, we add instrumental effects with the following three steps:

(i) *Mean removal*: Remove the mean of the signal for each frequency slice
(ii) *Instrumental noise*: Compute the *uv* coverage and sample a realization of the instrumental noise for a 1000 h measurement with SKA1-low
(iii) *Wedge removal*: Remove from the image all Fourier modes residing in a foreground-contaminated 'wedge'.

The first step is simply the result of measuring the signal with an interferometer, thus losing the global signal (we label it as $\delta\widetilde{T}_b$). The last two steps are further detailed below.

2.2.1 *Instrumental noise*

For our archetypal 21 cm interferometer we use SKA1-low, whose design is optimized for high signal-to-noise (S/N) images (Dewdney et al. 2013). The *uv* coverage and instrumental noise are calculated using `tools21cm`[3] (Giri, Mellema & Ghara 2018b; Giri, Mellema & Jensen 2020). We assume a tracked scan of $t_{\rm daily} = 6\,{\rm h\,d^{-1}}$, $t_{\rm int} = 10$ s integration time, and a total $t_{\rm obs} = 1000$ h measurement. For computational convenience, we fix the *uv* grid to the Fourier dual grid of the lightcone and use a box-car as a gridding kernel. With this approximation, the thermal noise is computed for each frequency slice based on the total time spent in each uv cell-by-all baselines throughout the 1000 h measurement (accounting for Earth's rotation):

$$\sigma_{uv} = \frac{T_{\rm sys} \cdot \Omega_{\rm beam}/\Omega_{\rm pix}}{\sqrt{2\Delta\nu\, t_{\rm int}}} \cdot \frac{1}{\sqrt{N_{uv} \cdot t_{\rm obs}/t_{\rm daily}}}\ [{\rm mK}], \tag{3}$$

[2] https://github.com/andreimesinger/21 cmFAST

[3] https://github.com/sambit-giri/tools21cm

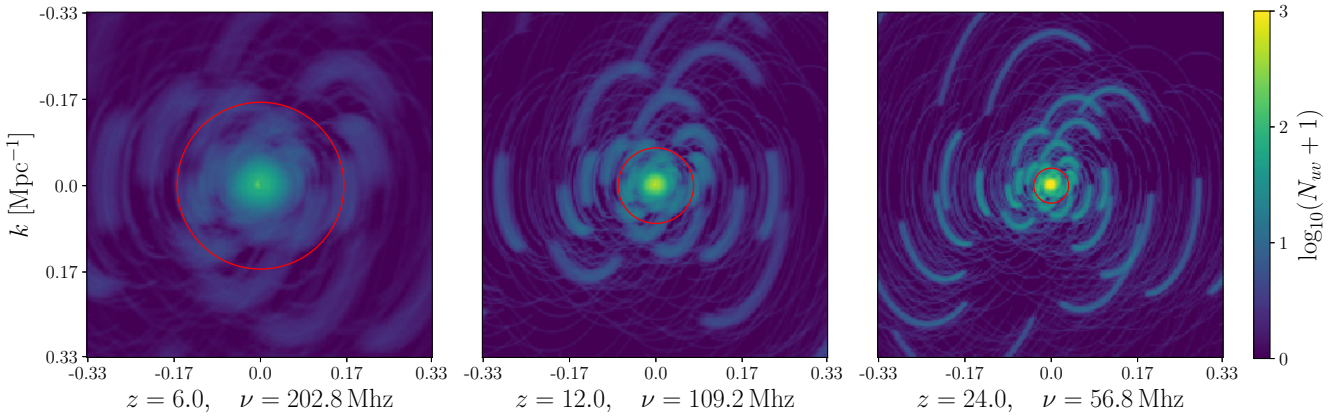


Figure 2. Daily *uv* coverage at different redshifts (frequencies). The red circle denotes the maximum baselines considered when computing the noise (2 km). $N_{\rm uv}$ represents the number of measurements in a given *uv* cell, for $t_{\rm obs} = 6$ h, $t_{\rm int} = 10$ s.

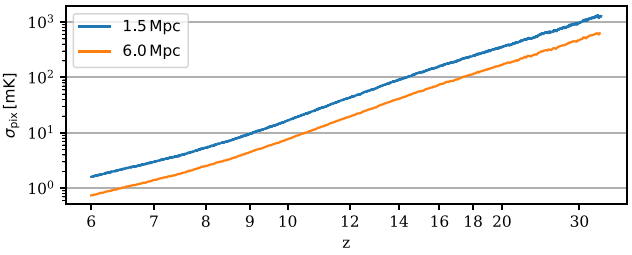


Figure 3. Telescope noise in mK at a pixel level as a function of redshift for our fiducial 1000h SKA observation. The two curves correspond to pixels of side-lengths 1.5 and 6 Mpc.

where the first factor on the RHS corresponds to the noise over a Fourier-cell from a single baseline, expressed in temperature units (e.g. Parsons et al. 2012), and the second factor accounts for multiple measurements of a given *uv* cell: $N_{\rm uv}$ integration times per night for $t_{\rm obs}/t_{\rm daily}$ nights. Thus, total observation time is discretized in visibility snapshots $t_{\rm int}$ apart. The effective beam solid angle, $\Omega_{\rm beam} \sim \lambda^2/A_{\rm eff} \approx 0.004(\nu/150\,{\rm MHz})^{-2}$ sr encodes the collecting area of the instrument, while the pixel solid angle accounts for the fact that the noise is inherently an angle-integrated quantity. $T_{\rm sys}$ for the SKA is given by

$$T_{\rm sys} = 60 \left(\frac{\nu}{300\,{\rm MHz}}\right)^{-2.55} \quad [{\rm K}]. \tag{4}$$

A realization of the observed, gridded visibilities including thermal noise is computed as $\delta\widetilde{T}_b(\boldsymbol{u}, \nu)_{\rm obs} = \delta\widetilde{T}_b(\boldsymbol{u}, \nu)_{\rm cosmo} + \mathcal{N}(\mu = 0, \sigma = \sigma_{uv})$, where $\delta\widetilde{T}_b(\boldsymbol{u}, \nu)_{\rm cosmo}$ is the simulated brightness temperature Fourier-transformed in the sky-plane (using the Discrete Fourier Transform convention in which no length-normalization is applied), and $\mathcal{N}$ is a random variable drawn from a zero-mean Gaussian distribution with variance σ^2_{uv}.

In Fig. 2, we show the *uv* coverage for a daily, 6 h observation, at different redshifts. When computing the thermal noise in equation (3), we use only the core stations: baselines shorter than 2 km, marked with a red circle, that provide most of the S/N. We additionally only consider cells with $N_{\rm uv} \geq 15$, roughly amounting to one full day of observation. For simplicity, our frequency bins match the native resolution of the cosmological simulation, $\Delta\nu(z) = 1.5$ Mpc.

In Fig. 3, we show the calculated thermal noise as a function of redshift, at two different spatial scales. Assuming the intrinsic signal has an rms of order 10s of mK on these scales, we expect good signal-to-noise images up to $z \lesssim 10$ and noise-dominated images from $z \gtrsim 15$.

2.2.2 *Wedge removal*

Foregrounds represent one of the largest obstacles to detecting the 21 cm signal. Efforts generally focus on mitigating the foregrounds or discarding the Fourier modes expected to be dominated by foregrounds (see e.g. Kerrigan et al. 2018; Chapman & Jelić 2019). Here, we take the latter, conservative approach: excising a foreground-dominated 'wedge' (in 2D cylindrical k-space) from the 21 cm lightcone.

An approximate relation for the contaminated wedge region (e.g. Morales et al. 2012; Vedantham, Udaya Shankar & Subrahmanyan 2012; Trott, Wayth & Tingay 2012; Parsons et al. 2014; Liu, Parsons & Trott 2014a, b; Murray & Trott 2018) can be derived based on a baseline's response to a foreground point-source:

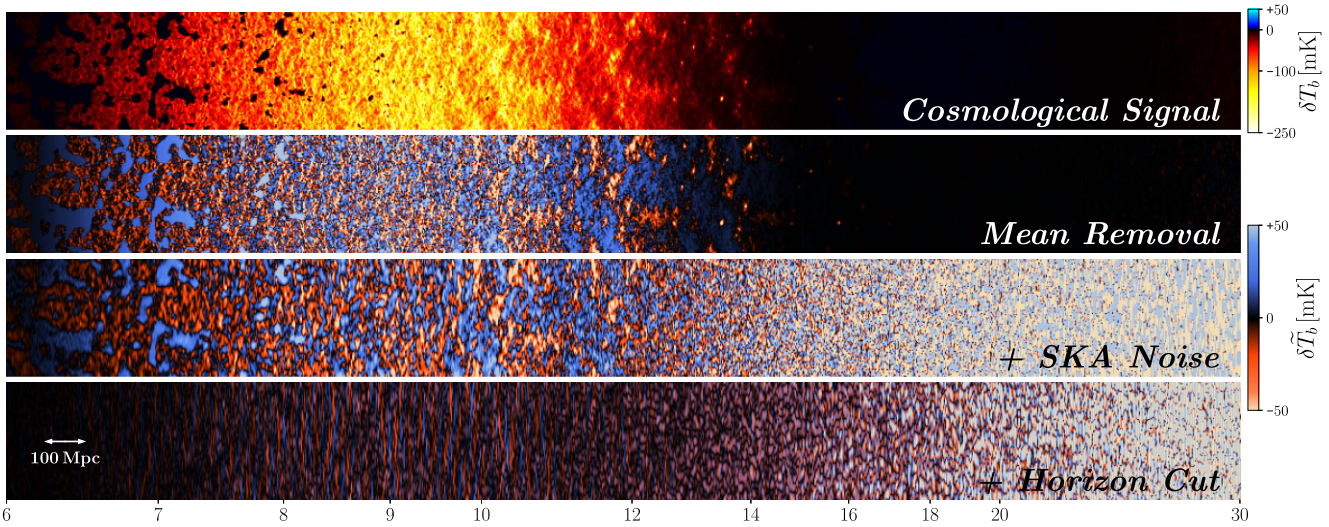
$$k_\parallel \leq \kappa(\boldsymbol{k}_\perp, z) \equiv |\boldsymbol{k}_\perp| \frac{E(z)}{1+z} \int_0^z \frac{{\rm d}z'}{E(z')} \cdot \sin\theta + b, \tag{5}$$

where $E(z) = \sqrt{\Omega_m(1+z)^3 + \Omega_\Lambda}$, and $\boldsymbol{k}_\perp$ are line-of-sight scales. Here, the additive term b captures the fact that at low-$k_\perp$, the width of the foregrounds in Fourier space is constant, and set by an applied 'frequency-taper'. We discuss this in more detail below. Finally, θ represents the zenith-angle of the point source. Since point-sources cover the sky, θ is defined by the viewing angle of the telescope. Estimates of θ can range from the full-width at half-maximum of the telescope beam as an optimistic choice to the full horizon as a pessimistic choice (cf. Pober et al. 2014). We again make a conservative choice and assume the horizon limit, $\theta = \pi/2$. Setting all Fourier modes that obey relation 5 to zero in principle removes all foregrounds (while at the same time removing some of the cosmic signal), and is known as 'wedge removal'.

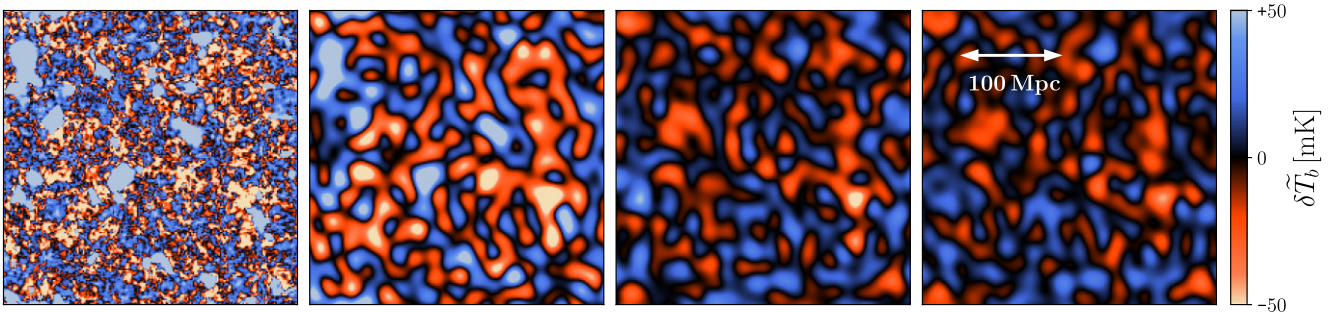
The wedge definition (equation 5) is inherently redshift-dependent, and thus cannot be applied to the Fourier-transform of an entire lightcone, as the redshift evolves along the lightcone. To consistently define the wedge as a function of redshift, we implement a rolling procedure. For each redshift slice at comoving distance $r_\parallel$, we

(i) take the part of the lightcone in range $r_\parallel \pm \Delta r/2$, where $\Delta r = 750$ Mpc,

(ii) multiply by Blackman–Harris (BH) taper function in the line-of-sight direction,



(a) Slices through the frequency axis of an example 3D lightcone, following successive contamination of the signal. From top to bottom: *Cosmological Signal*, *Mean Removal*, *Mean Removal + SKA noise*, *Mean Removal + SKA noise + Horizon Cut*.



(b) Slices trough the sky-plane at $z = 8$, corresponding to the example in (a). From left to right: *Mean Removal*, *Mean Removal + SKA noise*, *Mean Removal + SKA noise + Optimistic Wedge* ($\theta = 50°$), *Mean Removal + SKA noise + Horizon Cut* ($\theta = 90°$).

Figure 4. (a) Slices through the frequency axis of an example 3D lightcone, following successive contamination of the signal. From top to bottom: *Cosmological Signal*, *Mean Removal*, *Mean Removal + SKA noise*, *Mean Removal + SKA noise + Horizon Cut*. (b) Slices trough the sky-plane at $z = 8$, corresponding to the example in (a). From left to right: Mean Removal, Mean Removal + SKA noise, Mean Removal + SKA noise + Optimistic Wedge ($\theta = 50°$), Mean Removal + SKA noise + Horizon Cut ($\theta = 90°$). Steps in our observational pipeline. We generate a data base following each of the above three steps: *Mean Removal*, *+SKA Noise*, and *+Horizon Cut*. In panel (a), we show slices along the frequency plane, while in panel (b) we show slices in the sky-plane. In panel (b), we also include a more optimistic wedge contamination. Astrophysical parameters of the particular example are $\zeta = 150$, $T_{\rm vir} = 10^{5.7}$ K, $L_X/{\rm SFR} = 10^{40}$ erg s^{-1} M$_\odot^{-1}$ yr, $E_0 = 1.2$ keV.

(iii) Fourier-transform to 3D $\boldsymbol{k}$-space and apply wedge removal,
(iv) transform back to real space and save values for the central slice only.[4]

Here, the BH taper is employed to reduce artificial ringing in Fourier space, which would otherwise occur when applying a simple box-car window function (i.e. just selecting a 'chunk' of the lightcone) on a non-periodic boundary (e.g. Choudhuri et al. 2016; Trott et al. 2016). Such structure is inimical to the removal of foregrounds via the wedge, as the wedge relation (equation 5) itself depends on a highly compact Fourier Transform of the foregrounds over the line of sight. High-$k_{||}$ ringing due to the window function leaks foreground power outside the wedge, reducing this leakage.

However, applying the BH taper is not without trade-offs. While it reduces Fourier-space sidelobes out to high-$k_{||}$, it increases the width of the main foreground lobe at low-$k_{||}$. This results in the buffer b, which does not evolve with perpendicular scale, but represents the minimum $k_{||}$ at any perpendicular scale for which the data is foreground-free. This parameter increases for more compact taper functions (in frequency space), and similarly for smaller 'chunks' of the lightcone (i.e. $\propto \Delta r^{-1}$). We define b as the width in Fourier space where the dynamic range of the taper is 10^{-10}. We found that $\Delta r = 750$ Mpc optimizes between these trade-offs, but the results of this paper are not highly sensitive to the choice of it.

[4]If $M(\boldsymbol{r})$ is the measured data, described rolling procedure amounts to

$$M^*(\boldsymbol{r}_\perp, r'_\parallel) = \widetilde{\mathcal{F}}_{\boldsymbol{r}} \left\{ W(\boldsymbol{k}, r'_\parallel) \cdot \mathcal{F}_{\boldsymbol{r}} \left\{ B_N(r_\parallel - r'_\parallel) \cdot M(\boldsymbol{r}) \right\} \right\} \Big|_{r'_\parallel},$$

where $\mathcal{F}_{\mathbf{r}}$ and $\widetilde{\mathcal{F}}_{\mathbf{r}}$ are Fourier and inverse Fourier transforms, W is the wedge window function computed from equation (5), B_N is the Blackman-Harris of size $N = \Delta r/1.5$ Mpc centred around $r'_\parallel$.

2.2.3 Data set preparation

Using the 10 000 cosmological lightcone simulations described in Section 2.1, we generate three data bases, one for each step in the observational pipeline outlined in Section 2.2. From hereon, we refer to these three data bases as

(i) *Mean removal*
(ii) *+SKA noise*
(iii) *+Horizon cut*.

Each data base is randomly split as 80 per cent: 10 per cent:10 per cent into training:validation:test sets. In Fig. 4, we show an example 3D lightcone image [with cuts along the frequency (4a) and sky-plane axes (4b)] resulting after each step of the signal contamination. Removing the mean and adding SKA noise does not remove the large-scale features from the cosmological signal. However, removing the modes from the wedge does contaminate the image in a way that is difficult to decipher by eye. Below we quantify how well the NN recovers astrophysical parameters following each stage of contamination. Note that the wedge removal step could be considered pessimistic; current instruments like LOFAR and MWA are constantly improving foreground mitigation within the wedge (Li et al. 2018; Barry et al. 2019; Mertens et al. 2020; Hothi et al. 2021). Moreover, NNs could be trained to add the missing, foreground-contaminated modes (Gagnon-Hartman et al. 2021). For a comparison, we calculate the wedge excision in a more optimistic case (see Fig. 4b, where $\theta = 50^\circ$ is consistent with Mertens et al. 2020), however we do not use it in the training of NNs.

For data augmentation, each lightcone is separated into 4 smaller sky-plane patches and 10 realizations of noise are calculated per lightcone (see e.g. Perez & Wang 2017; Shorten & Khoshgoftaar 2019). For computational efficiency, we downsample the 3D images by a factor of 4 (to voxels of 6 Mpc), noting that this is below the expected SKA1-low resolution. Finally, to improve the stability in training, we normalize both the lightcone images and the parameter ranges to zero-mean and unit-variance (e.g. LeCun et al. 2012).

3 NETWORK ARCHITECTURES

Here, we describe the different network architectures we use for parameter estimation. These are based on combinations of convolutional and recurrent operations, guided by our specific usage case. Namely, the cosmic 21 cm signal contains spatially correlated information in the sky-plane and spatio-temporally correlated information across frequency bins. These correlations, intrinsic to the cosmic signal, are however weakened with increased data contamination from observational pipelines.

With this in mind, we construct three NN architectures – all using (primarily) 2D convolutions to encode local sky-plane correlations and either recurrent or 1D convolutional operations to encode information along the frequency dimension. Although CNNs can encode correlations in both sky and frequency planes, they treat the data as a stationary image. On the other hand, RNNs are designed to encode a sequence of data by 'rolling over' it and reusing the same weights on each step (see Appendix A for more details). One RNN layer then effectively becomes an NN with the depth equal to the length of a sequence – able to encode highly non-linear data by having comparably fewer weights. This design allows RNNs, and especially Long Short Term Memory RNNs (Hochreiter & Schmidhuber 1997), to efficiently and quickly find a stable local minimum of the loss function when training.[5] As a result, RNNs became famous in audio/video encoding (e.g. Shi et al. 2015; Zhao, Li & Lu 2019) and natural language processing (e.g. Aharoni, Rattner & Permuter 2017). Here, we introduce them to the field of 21 cm.

[5]Global minima of loss functions are almost impossible to find given the high dimensionality of the parameter space of deep network weights; however, local minima can result in satisfactory, comparable performance (e.g. Choromanska et al. 2015).

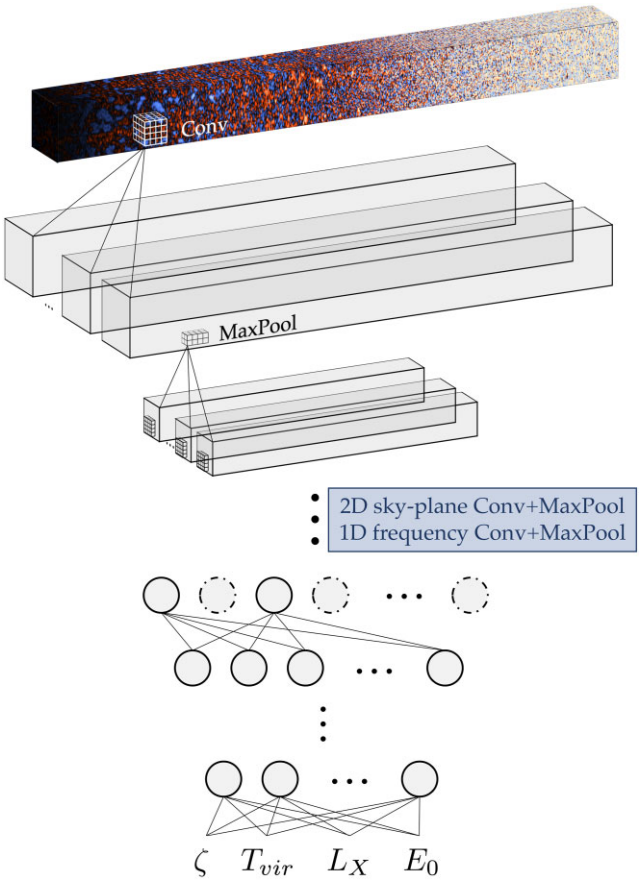


Figure 5. CNN architecture sketch. Parallel lightcones denote multiple filters and subsequent convolutional and pooling layers are depicted in blue. More details are found in Appendix B.

Below, we briefly sketch the specific CNN and RNN architectures we use in this study. Detailed descriptions, including the number of layers, filters, etc., can be found in Appendix B.

3.1 CNN

Fig. 5 shows a sketch of our CNN architecture. The convolutional part consists of iterative convolutional ('Conv') and pooling ('MaxPool') layers, and is followed by fully connected (FC) layers. The final output is the prediction of the four astrophysical parameters we use in this study. 'Neurons' are shown with circles, where dash–dotted circles depict dropout at the first FC layer. Convolutions locally correlate voxels of the lightcone and pooling layers downsample it by keeping only the strongest activations.

In the first convolutional + pooling layer, we use 3D kernels. Subsequently, we iterate successive layers of 2D convolutions in the sky-plane and 1D convolutions across frequency bins. Two such layers combined effectively make a 3D convolutional layer, however with a reduced number of weights.

One advantage of CNN architectures is that they are comparably simple, capable of being trained using more modest computational resources. Our CNNs on average require 0.13 GPUh (NVIDIA P100) per epoch of training, and 20 ms per execution once trained.

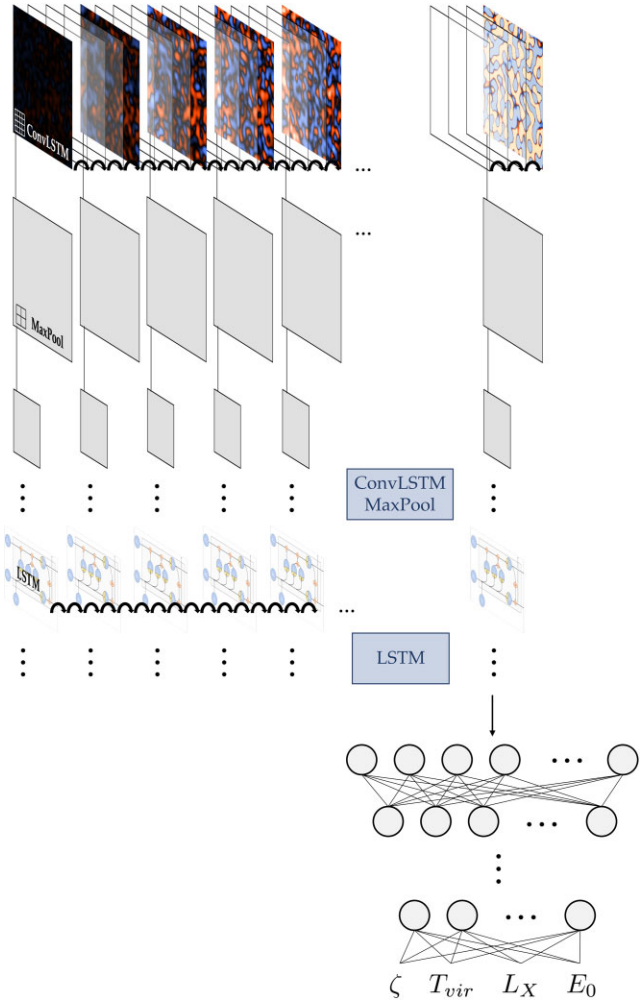


Figure 6. ConvRNN architecture sketch. For legibility, multiple filters/channels are omitted from the sketch. Curly arrows represent activations passing sequentially in the LSTM layer. Consecutive layers are depicted in blue. More details are found in Appendix B.

3.2 ConvRNN

Fig. 6 shows a sketch of our ConvRNN architecture. The input lightcones are passed through a series of convolutional Long Short Term Memory (ConvLSTM; Shi et al. 2015) layers, which combine 2D convolutions in the sky-plane with recurrent LSTMs in the frequency dimension. As for the CNN, the dimensionality is reduced following each convolution with a MaxPool layer. The ConvLSTM + MaxPool layers are then followed by pure LSTM and finally FC layers, leading to the parameter predictions.

Despite their efficient performance, a notable drawback is the requirement of substantial computational resources. Our ConvRNNs on average require 2 GPUh (NVIDIA P100) per epoch of training, and 6 s per execution once trained. This is a factor of 15 larger in the training time compared with the CNN described above. Because of these substantial computational resources, we also implement a 'slimmed-down' RNN that we refer to as 'SummaryRNN' below.

3.2.1 SummaryRNN

The main computational bottleneck of the ConvRNN is the convolutional back propagation through time (e.g. Werbos 1990). Therefore, we also construct a 'SummaryRNN', in which we remove all recurrent operations from the convolutions, replacing the ConvLSTM layers in Fig. 6 with pure 2D sky-plane convolutions. Thus, the SummaryRNN first 'summarizes' the lightcones using sequential sky-plane convolutions, and only passes these (downsampled) summaries into a stack of LSTM and FC layers. The SummaryRNN architecture sketch would effectively be identical to the ConvRNN in Fig. 6, but without the curved arrows (representing recurrent operation) in the upper row.

Although the resulting SummaryRNN loses some information compared with the ConvRNN, it can train considerably faster. Specifically, our SummaryRNNs take 0.1 GPU/h (NVIDIA P100) per epoch, and 45 ms to execute. This is a factor of 20 improvement in training time compared with ConvRNN.

Table 1. Hyperparameter space explored with a grid search. Training was performed using a reduced number of epochs for various hyperparameter combinations. The final choices are indicated in red. For more details on the network architectures, see Appendix B.

Batch size	20, 100
Initial learning rate	10^{-2}, 10^{-3}, 10^{-4}, 10^{-5}
Dropout	0.2, 0,5
Batch normalization	`True`, `False`
Optimizer	`RMSprop`, `SGD`, `Adamax`, `Adam`, `Nadam`
Activation function	`relu`, `leakyrelu(0.1)`, `elu`, `selu`

4 TRAINING

4.1 General network set-up

We make use of several standard techniques to improve the stability and generalization of the NN training, including optimal initialization of the network, batch normalization and dropout. For each choice of activation function, we use the corresponding optimal initialization schemes identified in Lecun, Bengio & Hinton (2015) and He et al. (2015) on the basis of keeping the variance of the weights constant during the training. The activation function is applied to all hidden layers of the network. However for LSTMs, we keep the internal structure and activations unchanged (for details see Fig. A1). Dropout (Srivastava et al. 2014) is applied only once, in the widest part of every NN. The performance of the network is evaluated by calculating the mean square error (MSE) between the true and predicted values of the parameters. The weights are updated with back-propagation using a fixed batch size (Rumelhart, Hinton & Williams 1986). Batch normalization is applied immediately before any tensor transformation, with an exception of a few layers at the very end (Ioffe & Szegedy 2015).

We perform a grid search of standard NN hyperparameter combinations, listed in Table 1. For this, we train each NN over a reduced number of epochs (100) and use the resulting MSE loss to identify the best hyperparameters. This required ≈500 trainings per architecture and data base. Hyperparameters marked with red in Table 1 are preferred by all combinations of NN architectures and data bases. In general, the performance was extremely sensitive to the initial learning rate. We notice better performance for lower batch size and dropout, while still allowing the final network to generalize well (as we shall see in the next section). Turning on batch normalization made the training more stable, while we saw no difference between

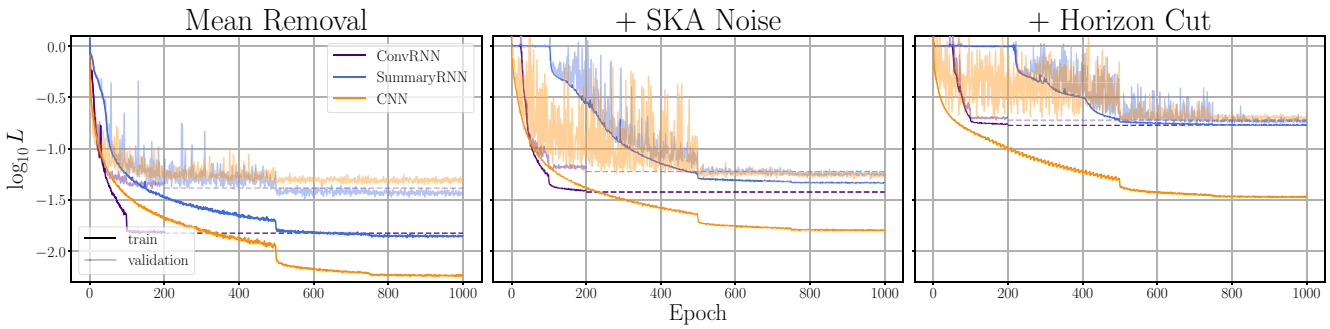


Figure 7. Training and validation losses for the final NN architectures. CNN, ConvRNN, and SummaryRNN architectures are depicted in orange, purple, and blue, respectively. The panels correspond to our three data bases, from left to right in increasing levels of signal contamination. ConvRNN converges extremely quickly and we train it for 200 epochs only (dashed lines denote the final loss after 200 epochs). The other NNs were trained for 1000 epochs.

`Adam` and `Nadam` optimizers, and between `relu` and `leakyrelu`[6] activations. Final architectures are then trained until convergence (for a detailed specification, we refer the reader to Appendix B).

The training was done in parallel on 10 NVIDIA P100 GPUs with the `ring-Allreduce` update scheme. For this, we used `TensorFlow`[7] (Abadi et al. 2015) as a main framework and `horovod`[8] (Sergeev & Del Balso 2018) as a parallelization library. When training on multiple GPUs, a copy of an NN is held on each device and the final loss is an average across all individual losses. Thus, with the constant batch size (bs) per device, the effective batch size grows with the number of GPUs ($N_{\rm GPU}$) as $\mathrm{bs}_{\rm eff} = N_{\rm GPU} \cdot \mathrm{bs}$. To cope with this, in the first 10 epochs we linearly scale the learning-rate $\mathrm{lr} \rightarrow \mathrm{lr} \cdot N_{\rm GPU}$ [so-called warmup, see Goyal et al. (2017) for details].

Finally, for a consistent comparison, we fixed the learning rate scheduler – reducing it by a factor of 10 on 50 per cent and 75 per cent of the training.

4.2 Training performance

In Fig. 7, we show the training and validation losses for our three network architectures. The panels correspond to our three data bases: (i) *Mean removal*; (ii) + *SKA noise*; and (iii) + *Horizon cut*, from left to right. For the *Mean removal* data base, we see that both RNNs outperform the CNN. This is evidenced mainly by their validation losses being lower. Furthermore, the difference between the training and validation losses is much smaller for the two RNNs than for the CNN, demonstrating that the RNNs are able to generalize better and are less prone to overfitting.

We see that the final validation losses of ConvRNN and SummaryRNN are comparable for the *Mean removal* data base. However, the ConvRNN (which includes recurrent layers also in the convolutional steps) is much more stable in training. It rapidly and with very little stochasticity finds a local minimum in the loss function, after only 100 epochs; SummaryRNN requires 500 epochs to approach a comparable loss and the training is noisier initially (note the effect of LR reduction at 50 per cent and 75 per cent of the training). However, even accounting for ∼5 times more training epochs, SummaryRNN still is less computationally intensive compared to ConvRNN (a factor of 4 fewer GPUh in total training time).

[6]Here we use $\texttt{leakyrelu(0.1)} = \begin{cases} 0.1\,x & x < 0 \\ x & x \geq 0 \end{cases}$.

[7]https://github.com/tensorflow/tensorflow

[8]https://github.com/horovod/horovod

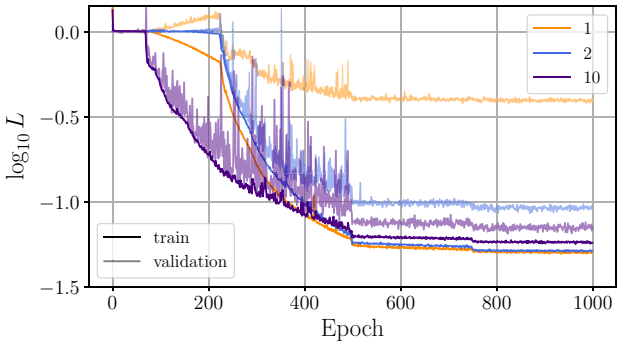


Figure 8. Evolution of the training and validation losses for SummaryRNN when varying the number of noise realizations per cosmic signal in the + *SKA noise* data base. The fiducial data bases used here include 10 realizations per cosmic signal. We see that the validation loss keeps decreasing as more noise realizations are included. This is suggestive that our final NN performance is limited by our data augmentation (data base size).

From the middle and right-hand panels of Fig. 7, we see that with higher contamination, the performance worsens and the differences in the final validation losses between the architectures disappear. The fact that the different architectures are reaching the same validation loss is strongly suggestive that we are reaching the intrinsic limits of our data sets.

To explore this point further, we vary the number of noise realizations per cosmic signal in the + *SKA noise* database, re-training the SummaryRNN each time. In Fig. 8, we show the training and validation losses using 1, 2, and 10 noise realizations per cosmic signal (our fiducial data base corresponds to 10). We note a significant decrease in the validation loss going from 1 to 2 noise realizations per cosmic signal. However, although the improvement is smaller, the final validation loss keeps decreasing even down to 10 realizations. This supports the claim above that our results are limited by our data augmentation, especially for the + *SKA Noise* and + *Horizon cut* data bases. In future work, we will increase the size of the data bases, sampling more cosmic signals and contamination realizations, quantifying if we reach convergence.

5 PARAMETER RECOVERY

In this section, we demonstrate the performance of the trained networks in parameter recovery on the test sets. We begin by showing the predicted versus true distributions, $P(p, t)$, for the SummaryRNN in Fig. 9. The rows correspond to our four astrophysical parameters,

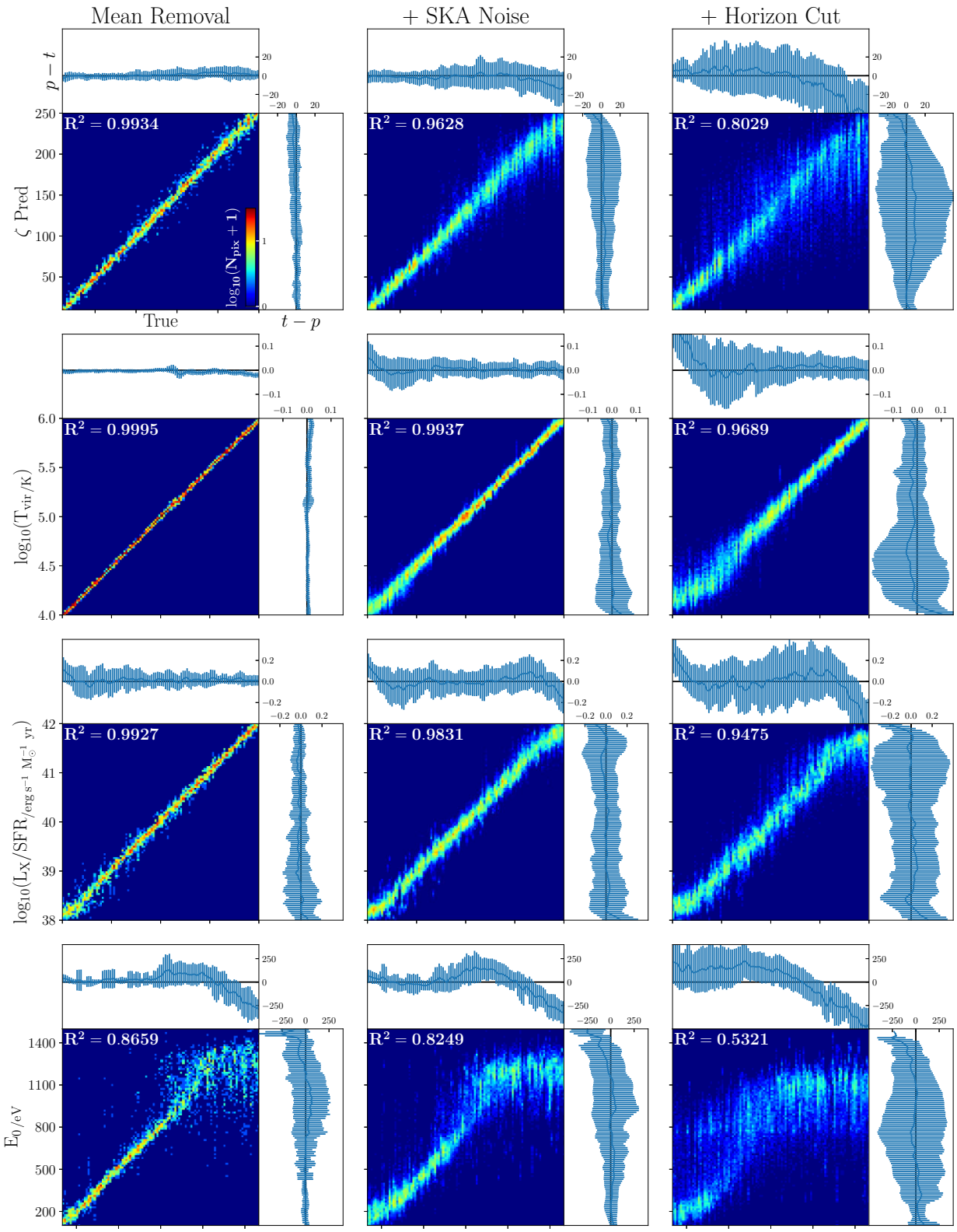


Figure 9. Histograms of True versus Predicted, $P(p, t)$, for our SummaryRNN architecture and all data bases. From left to right, columns represent: *Mean Removal*, *+SKA noise*, *+Horizon Cut* data sets. Rows correspond to our four astrophysical parameters (defined in the text). Each panel is bordered by the mean and $\pm 1\sigma$ of the 1D conditional distributions, $P(t - p|p)$ and $P(p - t|t)$. R^2 coefficients are marked in upper-left corners. Some labels are omitted to avoid overcrowding; see the top left-hand panel for the relevant labels.

while the columns correspond to the three different data bases. Standard R^2 scores are reported in the panels. On the top and side of each 2D distribution, we show the mean and $\pm 1\sigma$ of the 1D conditional distributions, $P(t-p|p)$ and $P(p-t|t)$, respectively.

Focusing on the *Mean Removal* results, we recover the same qualitative trends discussed in Gillet et al. (2019), who used the same database of cosmic signals. The best recovery is obtained for the minimum virial temperature for star-forming galaxies, $T_{\rm vir}$, with an $R^2 = 0.9995$ and no notable biases. This is because $T_{\rm vir}$ impacts the timing of *all* astrophysical epochs, as well as the characteristic scales of structures. Thus the cosmic signal is very sensitive to $T_{\rm vir}$, facilitating good recovery. The ionizing efficiency and X-ray luminosity parameters, ζ and L_X/SFR, are also predicted very well. These impact the timing of the EoR and the EoH, respectively. The minimum X-ray energy escaping the host galaxy, E_0, is recovered well for values below $E_0 < 1$ keV. Since the interaction cross-section for X-rays is a strong function of energy, photons with higher energies are inefficient at ionizing / heating the IGM and do not leave a strong imprint in the 21 cm signal. As we approach $E_0 \rightarrow 1.5$ keV, corresponding to photons with mean free paths comparable to the Hubble length, the network prediction becomes understandably randomized (pulling towards the mean of the range). Indeed, all distributions show a characteristic 'S' shape, as a consequence of the sharp boundary on the parameter ranges.

As could be expected from the validation loss curves in the previous section, the SummaryRNN predictions on the test sets notably worsen with increasing signal contamination (going from left to right in Fig. 9). Including SKA noise[9] only decreases the R^2 scores by a few per cent. The wedge excision has a more dramatic effect, especially on the ζ and E_0 parameter predictions that drop to $R^2 = 0.80$ and 0.53, respectively.

After showing all of the predicted versus true distributions for the SummaryRNN architecture, we now compare the parameter recovery from all three of our architectures, using their $P(t|p)$.[10] In Fig. 10, we bracket $\pm 1\sigma$ (rms) of the $P(t-p|p)$ distributions for all architectures (identical to $P(t|p)$, only shifted for easier visualization), with upper and lower curves of matching colours. Columns represent different databases and rows different astrophysical parameters. In the case of unbiased errors, $P(t-p|p)$ would be a zero-mean distribution.

As expected from the validation loss curves, the two RNNs perform the best on the *Mean Removal* test data. Overall, SummaryRNN performs the best, likely outperforming ConvRNN due to the longer training (the number of training epochs for ConvRNN was five times less than for SummaryRNN, due to its substantial computational requirements, as discussed above). Specifically, SummaryRNN has MSE on average a factor of $\sim$2 lower than the CNN for all parameters. Here, the MSE is calculated for each parameter individually, averaging over all test samples. For detailed numerical values for all architectures and parameters, see Table 2.

In these *Mean Removal* panels of Fig. 10, we also show in grey the corresponding limits from the CNN presented in Gillet et al. (2019). Training on the same data base of cosmic signals, Gillet et al. (2019) considered a shallower CNN than we use here, and did not include any signal contamination (not even mean removal). Our deeper networks result in a factor of $\sim$2−8 smaller variance in $P(t|p)$ (calculated as an average across all test samples), despite having a factor of four times poorer resolution and having removed the mean of the cosmic signal.

For the + *SKA Noise* test data base, the prediction errors are larger for all parameters, and there are little differences between network performances (suggestive that we are limited by the data set size, as discussed previously). Some of the qualitative trends are understandable on physical grounds. As noted previously, for high values of E_0 that do not impact the signal the network predictions are almost randomly distributed across the whole range, resulting in the negative bias seen in the figure beyond $E_0 \gtrsim 1.2$ keV. Furthermore, for low values of $T_{\rm vir}$, the first galaxies form at very high redshifts, shifting all astrophysical epochs of the 21 cm signal to frequencies with higher thermal noise. This explains the (modest) increase in the prediction error at $T_{\rm vir} \lesssim 10^{4.3}$ K. The fact that the increase is relatively modest suggests all networks have learned to 'marginalize over' the noise reasonably well even with only 10 noise realizations per cosmic signal.

For the + *Horizon Cut* test data base, the prediction errors increase significantly for all architectures. This is not surprising given that our pessimistic wedge removal throws away a significant amount of information (c.f. Fig. 4). Unlike for the thermal noise that is uncorrelated with the cosmic signal, we cannot augment our + *Horizon Cut* data base without running more samples of the cosmic signal. As a result, all networks are unable to generalize and perform much worse than with the other data bases. In future work, we will investigate how well NNs can train to marginalize over simulated foregrounds, rather than adopting the simple foreground-avoidance approach as we do here (for a foreground-cleaning example, see La Plante & Ntampaka 2019).

5.1 What features are guiding the network predictions?

One major drawback of machine learning is that deep neural networks are often treated as 'black boxes'. As such, it is important to check if the trained NN is using reasonable features in the images to make predictions, and is not overfitting by focusing on unphysical artefacts particular to a data set (e.g. Lapuschkin et al. 2019). For this reason, feature identification tools such as saliency mapping and attention mechanisms are becoming increasingly popular (e.g. Zeiler & Fergus 2013; Selvaraju et al. 2016; Chang et al. 2018; Vaswani et al. 2017; Ramachandran et al. 2019).

Here, we use a simple saliency mapping technique (Simonyan, Vedaldi & Zisserman 2013) to visualise the features used by our CNN.[11] Specifically, we calculate a gradient saliency map, constructing a Jacobian matrix of the NN prediction with respect to the input image:

$$\mathcal{J}_0 = \left\langle \frac{\partial \boldsymbol{\theta^p}}{\partial \mathbf{d}}\bigg|_{\mathbf{d}_0} \right\rangle, \tag{6}$$

[9]We remind the reader that our noise calculation is done in *uv* space, and includes the effects of the finite beam.

[10]Although $P(p|t)$ is a common performance metric in the literature, it is less meaningful than $P(t|p)$. In practice, we will have an observation that when fed into a trained network will give a 'best guess' parameter vector, $\boldsymbol{\theta^p}$. So the relevant uncertainty is the probability of the 'true' parameters of the Universe, given this network prediction, $P(\boldsymbol{\theta^t}|\boldsymbol{\theta^p})$. Unfortunately, our point estimate networks do not allow for a direct calculation of the Bayesian posterior; however, the closest approximation we can make with the data at hand are the 'marginalized' distributions of $P(t|p)$ from the test set, where t and p are components of $\boldsymbol{\theta^t}$ and $\boldsymbol{\theta^p}$, respectively.

[11]Unfortunately, RNN visualization requires more complicated techniques (Karpathy, Johnson & Fei-Fei 2015; Ramanishka et al. 2017; Adel Bargal et al. 2018) that are not straightforward to adapt to our usage case. We thus use the CNN feature maps as a 'sanity check' in this section.

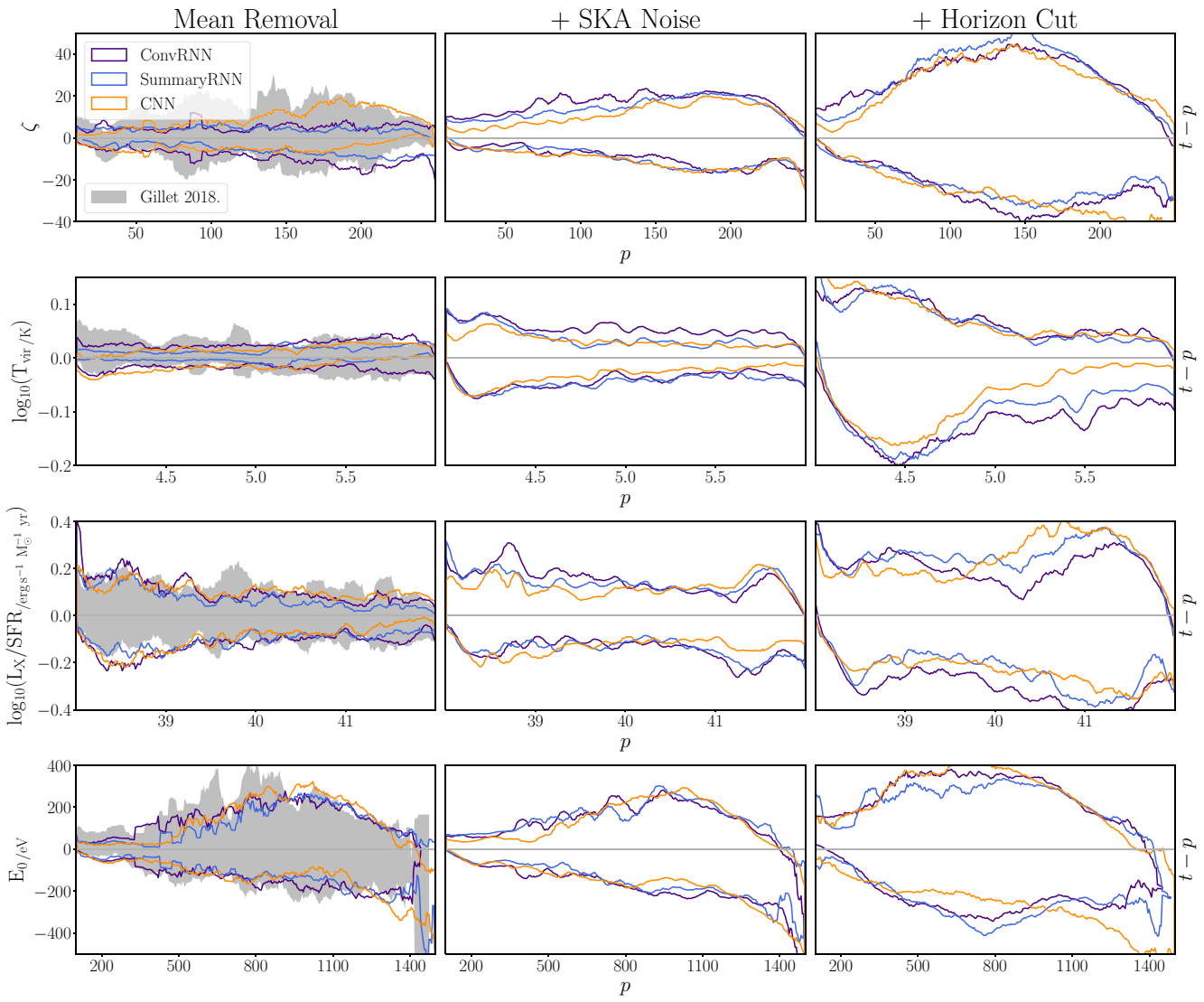


Figure 10. Prediction errors for all architectures. The lower and upper set of corresponding curves bracket $\pm 1\sigma$ of the $P(t-p|p)$ distributions. Columns represent different test data bases, and rows different astrophysical parameters.

Table 2. MSE of all considered models and parameters. All values are expressed in parameter units squared (e.g. E_0 in eV^2). ConvRNN, SummaryRNN, and CNN are marked with purple, blue, and orange, respectively.

	Mean rem.	+SKA noise	+Hor. cut
ζ	65 31 71	207 175 144	921 925 879
$log_{10}(T_{\mathrm{vir}})$	58×10^{-5} 16×10^{-5} 35×10^{-5}	260×10^{-5} 198×10^{-5} 125×10^{-5}	1175×10^{-5} 981×10^{-5} 749×10^{-5}
$log_{10}(L_X/\mathrm{SFR})$	15×10^{-3} 9×10^{-3} 12×10^{-3}	24×10^{-3} 22×10^{-3} 17×10^{-3}	77×10^{-3} 68×10^{-3} 64×10^{-3}
E_0	25×10^3 22×10^3 28×10^3	30×10^3 28×10^3 30×10^3	77×10^3 82×10^3 81×10^3

where $\boldsymbol{\theta}^{\boldsymbol{p}}$ is the NN parameter prediction vector in which each component corresponds to an astrophysical parameter, $\mathbf{d}$ is the data vector in which each component corresponds to a given pixel of the 3D lightcone image, $\mathbf{d}_0$ is a given input image for which we evaluate the saliency map, and the averaging is performed over all possible 150×150 Mpc sky-plane cuts out of the initial 300×300 Mpc lightcone. Besides giving us a consistent way to visualize a saliency map for the whole simulation volume, averaging smooths-out fluctuations in the gradients (for details about gradient smoothing, see Smilkov et al. 2017). Intuitively, the gradient $\partial\boldsymbol{\theta}^{\boldsymbol{p}}/\partial\mathbf{d}$ corresponds to the change in the NN parameter prediction from a pixel-by-pixel perturbation in the input image.

In Fig. 11, we use a sample input from our *Mean Removal* test set to illustrate the feature identification of our CNN. For simplicity, we only show 2D slices through the 3D lightcones. The top panel corresponds to the cosmic signal, the second panel to the input image, and the bottom four are the (normalized) gradient saliency maps for each parameter. These gradient saliency maps are also commonly referred to as 'heat maps', since brighter colours (larger

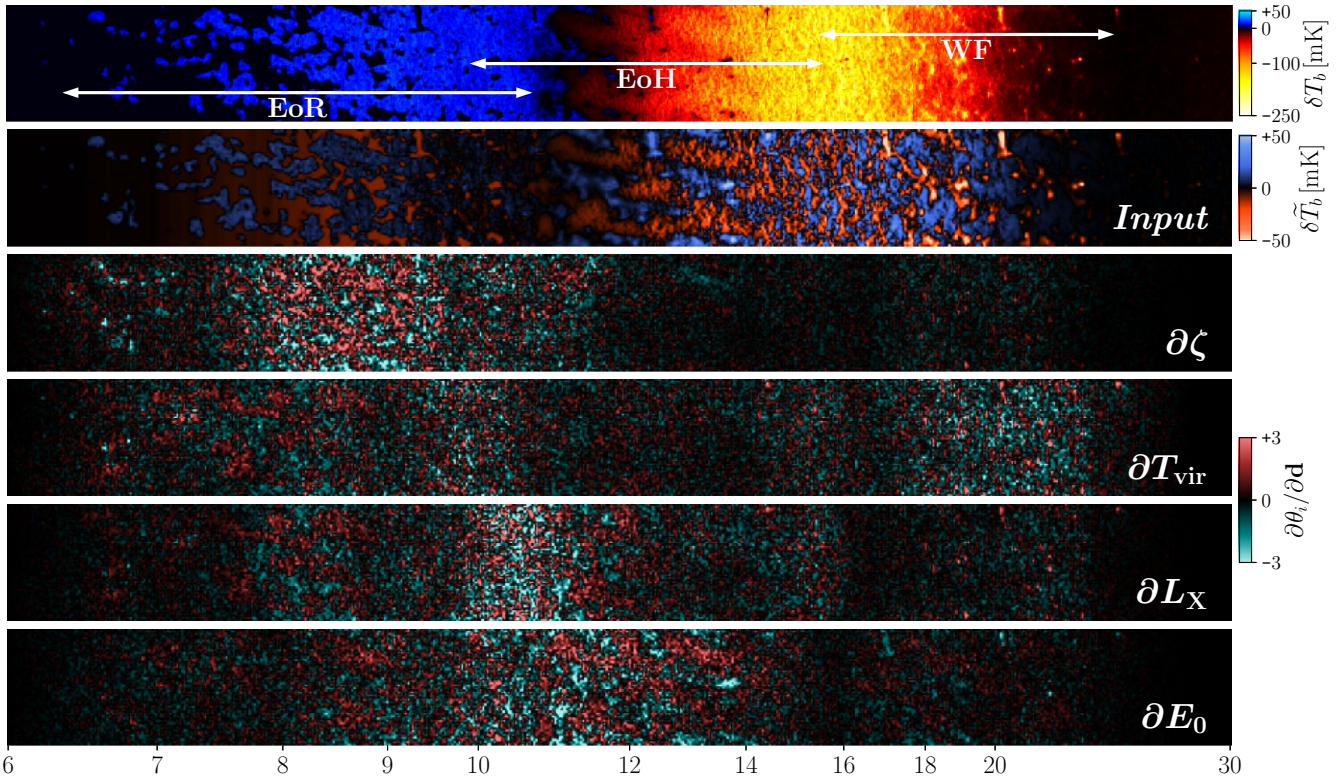


Figure 11. Gradient saliency maps for a sample image from the *Mean Removal* test set, constructed with our trained CNN. The top panel corresponds to the cosmic signal, the second panel to the input image, and the bottom four are the gradient saliency maps for the indicated parameter (normalized to have unit variance). For visualization purposes, we only show 2D slices through the 3D lightcones. The brighter colours (larger gradients) highlight the regions of the image that are important for each parameter prediction. For each parameter, the network correctly focuses on the relevant astrophysical epoch(s) of the 21 cm signal (denoted with arrows in the top panel). For reference, the cosmic signal used in this example corresponds to the following parameters: $\zeta = 44$, $T_{\rm vir} = 10^{4.7}$ K, $L_X/{\rm SFR} = 10^{40}\,{\rm erg\,s^{-1}\,M_\odot^{-1}\,yr}$, $E_0 = 0.6\,{\rm keV}$.

gradients) highlight the regions of the image that are important for each parameter prediction.

Unlike everyday images, our lightcones do not have 'objects' (cats, dogs, boats) that would make feature identification from heat maps straightforward. However, we can still draw some general conclusions from Fig. 11, by noting that the network has learned to correctly focus on the astrophysical epoch(s) relevant to each parameter (see the discussion of parameters in Section 2.1). For example, the minimum virial temperature for star-forming galaxies, $T_{\rm vir}$, impacts when galaxies form. Thus, it drives the timing of *all* astrophysical epochs: EoR, EoH, Wouthuysen-Field (WF) coupling (roughly demarcated with arrows for this example in the top panel of Fig. 11). Indeed, the gradient saliency map shows that the $T_{\rm vir}$ prediction is sensitive to the whole redshift range of the input image. Conversely, the ionizing efficiency, ζ, is mostly relevant for the timing of the EoR: the NN correctly focuses on this epoch, additionally using the HI patches remaining during the late stages ($z \sim 7$ for this model) for its prediction. Likewise, the X-ray heating parameters, L_X/SFR and E_0, are most sensitive to the EoH. While L_X/SFR mostly regulates the relative timing of the EoH, E_0 parametrizes the hardness of the emerging X-ray spectra and thus drives the relative sizes of the heated regions. Indeed, we see in the heat maps that E_0 is more sensitive to changes in the large, heated IGM structures.

We conclude therefore that the qualitative trends shown in Fig. 11 provide a good sanity check that our NN has learned physically relevant information and is not overfitting. Drawing more quantitative insights from gradient saliency maps is difficult, given that they only show parameter sensitivity to uncorrelated, pixel-by-pixel perturbations. Saliency maps using different basis sets (e.g. wavelets, superpixels) might be more useful for physically meaningful feature identification from 21 cm lightcones. We defer this to future work.

6 CONCLUSIONS

Upcoming images of the cosmic 21 cm signal will provide a physics-rich data, encoding both galaxy properties as well as physical cosmology (e.g. see the recent review in Mesinger 2019). Optimally extracting information from these lightcone images is challenging, since there is no obvious *a priori*, physically motivated optimal summary statistic. Since they are non-Gaussian, the common approach of compressing the images into a power spectrum could waste valuable information.

Here, we build on previous work (Gillet et al. 2019; La Plante & Ntampaka 2019; Kwon et al. 2020; Hortúa et al. 2020; Mangena et al. 2020) training NNs to predict astrophysical parameters directly from 21 cm images. We introduce RNNs to this field. RNNs are designed to characterize temporal evolution by passing through an image sequence with the same set of weights – allowing them to efficiently find local minima in the loss function (see Appendix A for more details).

To compare the performances of RNNs and traditional CNNs, we construct three data sets with increasing levels of signal contamination (c.f. Fig. 4): (i) *Mean Removal*, (ii) + *SKA Noise*, (iii) +

Horizon Cut. We train our NN architectures on all three databases using MSE minimization of the parameter predictions. We vary four astrophysical parameters to generate the underlying cosmic 21 cm signal, capturing the UV and X-ray properties of the first galaxies. These parameters were chosen as they are the most physically motivated 'tuning knobs' driving a large variation in possible signals.

We find the RNNs outperform CNNs on images with minimal signal contamination (our *Mean Removal* data base). The mean square prediction errors for the best RNN architecture, *SummaryRNN* were a factor of ∼2 lower than for a CNN of comparable depth, and up to a factor of ∼8 lower than the previous results applying a shallower CNN on the same database of cosmic signals (Gillet et al. 2019). It is important to note that these correspond to the prediction errors of point-estimate NNs. In future work, we will imply Bayesian techniques to obtain the full posterior over our parameter space (e.g. Hortúa, Malago & Volpi 2020; Zhao et al. 2021).

Using gradient saliency maps, we confirm that our NNs are identifying physically relevant features when trained. The networks focus on the correct astrophysical epoch(s) that are relevant for each parameter.

When trained on the signal contaminated images (+*SKA Noise* and +*Horizon Cut*), parameter prediction becomes less accurate. However, even in the most pessimistic case, parameters are predicted with reasonable accuracy (with R^2 ranging from 0.53 to 0.97). All architectures perform comparably on the contaminated images, which is likely due to the limited size of our data sets. Moreover, our foreground avoidance technique was fairly conservative; better calibration and foreground removal can improve parameter estimation. In future work, we will explore these trends further, quantifying the data set size needed for accurate parameter estimation, given different levels of image contamination.

ACKNOWLEDGEMENTS

This work was supported by the European Research Council (ERC) under the European Union's Horizon 2020 research and innovation programme (grant agreement no. 638809 – AIDA – PI: Mesinger). The results presented here reflect the authors' views; the ERC is not responsible for their use. We are grateful to PRACE for awarding us computational resources at Piz Daint, Switzerland, through the PRACE tier-0 grant AIfor21cm (project no. 2019204987). We acknowledge the CINECA award under the ISCRA initiative, for the availability of high performance computing resources and support (IsC80_MGLA21cm, project no. HP10CBAK8I). We also acknowledge computational resources of the HPC centre at SNS.

DATA AVAILABILITY

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

REFERENCES

Abadi M. et al., 2015, TensorFlow: Large-Scale Machine Learning on Heterogeneous Systems, https://www.tensorflow.org/
Adel Bargal S., Zunino A., Kim D., Zhang J., Murino V., Sclaroff S., 2018, Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), preprint (arXiv:1711.06778)
Aharoni Z., Rattner G., Permuter H., 2017, preprint (arXiv:1708.08863)
Barkana R., Loeb A., 2004, ApJ, 609, 474
Barry N. et al., 2019, ApJ, 884, 1
Bianco M., Giri S. K., Iliev I. T., Mellema G., 2021, MNRAS, 505, 3982
Chang C.-H., Creager E., Goldenberg A., Duvenaud D., 2018, preprint (arXiv:1807.08024)
Chapman E., Jelić V., 2019, preprint (arXiv:1909.12369)
Choromanska A., Henaff M., Mathieu M., Ben Arous G., LeCun Y., 2015, in Lebanon G., Vishwanathan S. V. N., eds, Proceedings of the Eighteenth International Conference on Artificial Intelligence and Statistics, 38. PMLR, San Diego, California, USA, p. 192
Choudhuri S., Bharadwaj S., Chatterjee S., Ali S. S., Roy N., Ghosh A., 2016, MNRAS, 463, 4093
Dewdney P., Turner W., Millenaar R., McCool R., Lazio J., Cornwell T., 2013, Document number SKA-TEL-SKO-DD-001 Revision, 1
Dumoulin V., Visin F., 2016, preprint (arXiv:1603.07285)
Furlanetto S. R., Zaldarriaga M., Hernquist L., 2004, ApJ, 613, 1
Furlanetto S. R., Oh S. P., Briggs F. H., 2006, Phys. Rep., 433, 181
Gagnon-Hartman S., Cui Y., Liu A., Ravanbakhsh S., 2021, MNRAS, 504, 4716
Gazagnes S., Koopmans L. V. E., Wilkinson M. H. F., 2021, MNRAS, 502, 1816
Ghara R. et al., 2020, MNRAS, 493, 4728
Gillet N., Mesinger A., Greig B., Liu A., Ucci G., 2019, MNRAS, 484, 282
Giri S., Mellema G., Jensen H., 2020, J. Open Source Softw., 5, 2363
Giri S. K., Mellema G., Dixon K. L., Iliev I. T., 2018a, MNRAS, 473, 2949
Giri S. K., Mellema G., Ghara R., 2018b, MNRAS, 479, 5596
Giri S. K., Mellema G., Aldheimer T., Dixon K. L., Iliev I. T., 2019, MNRAS, 489, 1590
Gorce A., Pritchard J. R., 2019, MNRAS, 489, 1321
Goyal P. et al., 2017, preprint (arXiv:1706.02677)
Greig B. et al., 2021, MNRAS, 501, 1
Greig B., Mesinger A., 2017, MNRAS, 472, 2651
Greig B., Mesinger A., 2018, MNRAS, 477, 3217
Hassan S., Andrianomena S., Doughty C., 2020, MNRAS, 494, 5761
He K., Zhang X., Ren S., Sun J., 2015, Proceedings of the IEEE International Conference on Computer Vision (ICCV), preprint (arXiv:1502.01852)
Hochreiter S., Schmidhuber J., 1997, Neural Comput., 9, 1735
Hortúa H. J., Malago L., Volpi R., 2020, Machine Learning: Science and Technology, 1, 035014
Hothi I. et al., 2021, MNRAS, 500, 2264
Ioffe S., Szegedy C., 2015, in Bach F., Blei D., eds, Proceedings of Machine Learning Research, Vol. 37, Proceedings of the 32nd International Conference on Machine Learning. PMLR, Lille, France, p. 448
Jennings W. D., Watkinson C. A., Abdalla F. B., McEwen J. D., 2019, MNRAS, 483, 2907
Jozefowicz R., Zaremba W., Sutskever I., 2015, in Bach F., Blei D., eds, Proceedings of Machine Learning Research Vol. 37, Proceedings of the 32nd International Conference on Machine Learning. PMLR, Lille, France, p. 2342, http://proceedings.mlr.press/v37/jozefowicz15.html
Karpathy A., Johnson J., Fei-Fei L., 2015, preprint (arXiv:1506.02078)
Kern N. S., Liu A., Parsons A. R., Mesinger A., Greig B., 2017, ApJ, 848, 23
Kerrigan J. R. et al., 2018, ApJ, 864, 131
Kwon Y., Hong S. E., Park I., 2020, J. Korean Phys. Soc., 77, 49
La Plante P., Ntampaka M., 2019, ApJ, 880, 110
Lapuschkin S., Wäldchen S., Binder A., Montavon G., Samek W., Müller K.-R., 2019, Nature Commun., 10, 1096
LeCun Y., Boser B., Denker J. S., Henderson D., Howard R. E., Hubbard W., Jackel L. D., 1989, Neural Comput., 1, 541
Lecun Y., Bengio Y., Hinton G., 2015, Nature, 521, 436
LeCun Y. A., Bottou L., Orr G. B., Müller K.-R., 2012, in Montavon G., Orr G. B., Müller K.-R., eds, Neural Networks: Tricks of the Trade: Second Edition. Springer Berlin Heidelberg, Berlin, Heidelberg, p. 9
Li W. et al., 2018, ApJ, 863, 170
Liu A., Parsons A. R., Trott C. M., 2014a, Phys. Rev. D, 90, 023018
Liu A., Parsons A. R., Trott C. M., 2014b, Phys. Rev. D, 90, 023019
Majumdar S., Pritchard J. R., Mondal R., Watkinson C. A., Bharadwaj S., Mellema G., 2018, MNRAS, 476, 4007
Makinen T. L., Lancaster L., Villaescusa-Navarro F., Melchior P., Ho S., Perreault-Levasseur L., Spergel D. N., 2021, JCAP, 4, 081
Mangena T., Hassan S., Santos M. G., 2020, MNRAS, 494, 600
Mellema G., Koopmans L., Shukla H., Datta K. K., Mesinger A., Majumdar S., 2015, in Advancing Astrophysics with the Square Kilometre Array (AASKA14). p. 10, preprint (arXiv:1501.04203)

Mertens F. G. et al., 2020, MNRAS, 493, 1662
Mesinger A., ed., 2019, The Cosmic 21-cm Revolution. IOP Publishing, p. 2514
Mesinger A., Furlanetto S., 2007, ApJ, 669, 663
Mesinger A., Furlanetto S., Cen R., 2011, MNRAS, 411, 955
Mesinger A., Ewall-Wice A., Hewitt J., 2014, MNRAS, 439, 3262
Mirocha J., Furlanetto S. R., Sun G., 2017, MNRAS, 464, 1365
Mondal R. et al., 2020, MNRAS, 498, 4178
Morales M. F., Hazelton B., Sullivan I., Beardsley A., 2012, ApJ, 752, 137
Murray S. G., Trott C. M., 2018, ApJ, 869, 25
Park J., Mesinger A., Greig B., Gillet N., 2019, MNRAS, 484, 933
Park J., Gillet N., Mesinger A., Greig B., 2020, MNRAS, 491, 3891
Parsons A., Pober J., McQuinn M., Jacobs D., Aguirre J., 2012, ApJ, 753, 81

Parsons A. R. et al., 2014, ApJ, 788, 106
Perez L., Wang J., 2017, preprint (arXiv:1712.04621)
Planck Collaboration VI, 2020, A&A, 641, A6
Pober J. C. et al., 2014, ApJ, 782, 66
Ramachandran P., Parmar N., Vaswani A., Bello I., Levskaya A., Shlens J., 2019, in Wallach H., Larochelle H., Beygelzimer A., d' Alché-Buc F., Fox E., Garnett R., eds, Advances in Neural Information Processing Systems, Vol. 32. Curran Associates, Inc.
Ramanishka V., Das A., Zhang J., Saenko K., 2017, Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), preprint (arXiv:1612.07360)
Rumelhart D. E., Hinton G. E., Williams R. J., 1986, Nature, 323, 533
Schmidt R. M., 2019, preprint (arXiv:1912.05911)
Schmit C. J., Pritchard J. R., 2018, MNRAS, 475, 1213
Selvaraju R. R., Cogswell M., Das A., Vedantam R., Parikh D., Batra D., 2016, preprint (arXiv:1610.02391)
Sergeev A., Del Balso M., 2018, preprint (arXiv:1802.05799)
Shi X., Chen Z., Wang H., Yeung D.-Y., Wong W.-k., Woo W.-C., 2015, preprint (arXiv:1506.04214)
Shimabukuro H., Yoshiura S., Takahashi K., Yokoyama S., Ichiki K., 2015, MNRAS, 451, 467
Shimabukuro H., Yoshiura S., Takahashi K., Yokoyama S., Ichiki K., 2017, MNRAS, 468, 1542
Shorten C., Khoshgoftaar T. M., 2019, J. Big Data, 6, 1
Simonyan K., Vedaldi A., Zisserman A., 2013, preprint (arXiv:1312.6034)
Smilkov D., Thorat N., Kim B., Viégas F., Wattenberg M., 2017, preprint (arXiv:1706.03825)
Sobacchi E., Mesinger A., 2014, MNRAS, 440, 1662
Srivastava N., Hinton G., Krizhevsky A., Sutskever I., Salakhutdinov R., 2014, J. Machine Learn. Res., 15, 1929
Trott C. M. et al., 2016, ApJ, 818, 139
Trott C. M., Wayth R. B., Tingay S. J., 2012, ApJ, 757, 101
van den Oord A. et al., 2016, preprint (arXiv:1609.03499)
Vaswani A., Shazeer N., Parmar N., Uszkoreit J., Jones L., Gomez A. N., Kaiser L., Polosukhin I., 2017, Advances in neural information processing systems, p. 5998
Vedantham H., Udaya Shankar N., Subrahmanyan R., 2012, ApJ, 745, 176
Watkinson C. A., Giri S. K., Ross H. E., Dixon K. L., Iliev I. T., Mellema G., Pritchard J. R., 2019, MNRAS, 482, 2653
Watkinson C. A., Greig B., Mesinger A., 2021, preprint (arXiv:2102.02310)
Werbos P., 1990, Proc. IEEE, 78, 1550
Zeiler M. D., Fergus R., 2013, preprint (arXiv:1311.2901)
Zel'Dovich Y. B., 1970, A&A, 500, 13
Zhao B., Li X., Lu X., 2019, Proceedings of the 25th ACM International Conference on Multimedia. Association for Computing Machinery, New York, NY, USA, p. 863
Zhao X., Mao Y., Cheng C., Wandelt B. D., 2021, preprint (arXiv:2105.03344)

APPENDIX A: CONVOLUTIONAL AND LSTM LAYERS

Here, we briefly present the general structure of convolutional (Conv), Long Short Term Memory (LSTM) layers, as well as the combination of the two (ConvLSTM).

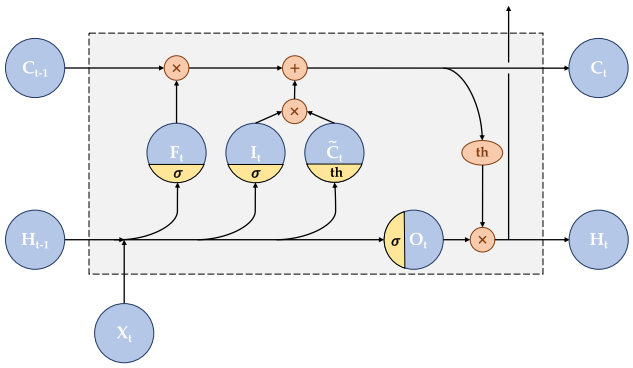


Figure A1. LSTM cell, showing the updating of a cell's states C_t and H_t for one time-step. Upper exiting arrow outputs H_t used for another LSTM layer if needed.

Let's start from a fully connected (FC) layer, for which we can write a vector transformation as

$$X_n = \psi(W_n^{n-1} \cdot X_{n-1} + B_n). \tag{A1}$$

Here, n represents a layer index, X_{n-1} and X_n are the input and output vectors, W_n^{n-1} and B_n the weight matrix and the bias vector, and ψ is the activation function.[12] It is easy to see from equation (A1) that every neuron takes the full input vector into account, hence the name of a FC layer.

However for an input containing spatially correlated data, it is more efficient to make a transformation which is in some sense local. The use of convolutions is the simplest choice. Each layer is represented by c^n convolutional filters and every 'channel' represents the output of one convolution. For the ith channel $(X_n)_i$, we can write

$$(X_n)_i = \psi\left(\sum_k (W_n^{n-1})_{ki} * (X_{n-1})_k + (B_n)_i\right), \tag{A2}$$

or in simplified notation: $X_n = \psi(W_{n-1}^n * X^{n-1} + B_n)$. The summation k is performed over c^{n-1} channels. In such way, the output $X_n(\mathbf{d})$ depends locally on X_{n-1} around $\mathbf{d}$, where locality is defined by a convolutional filter. For an intuitive and visual explanation of different convolutional operations, see the review by Dumoulin & Visin (2016).

Finally, let's consider the case of input X, where one of the dimensions (axis) can be considered as time (or in general a sequence). To incorporate that fact directly into layer design, we could compute for example 1D convolutions with respect to the time axis, where correlations are learned on a domain of the actual convolution (for a non-trivial example see van den Oord et al. 2016). However, we could also imagine encoding such information in an *iterative* manner. If we start with some 'hidden state' H_{t-1}, the operation

$$H_t = \psi(H_{t-1}, X_t; W, B) \tag{A3}$$

would take the input X_t and with already encoded information in H_{t-1}, update it into H_t. Such operations are the basis of RNNs.

Below we describe the specific choice of recurrent layers we use in our architectures: LSTM and ConvLSTM. Both are built on the general concepts mentioned above.

[12]For easier notation, we assume that ψ acts element-wise on its input.

A1 LSTM

Long Short Term Memory cells (Hochreiter & Schmidhuber 1997) solve several problems occurring in simple RNN structures. In particular, the problem of vanishing/exploding gradients is solved by separating short (fast) and long (slow) correlations in the data. In Fig. A1, we show a diagram of a LSTM cell, with the following accompanying equations:

$$\begin{aligned} F_t &= \sigma(W_X^F \cdot X_t + W_H^F \cdot H_{t-1} + B^F), \\ I_t &= \sigma(W_X^I \cdot X_t + W_H^I \cdot H_{t-1} + B^I), \\ \widetilde{C}_t &= th(W_X^C \cdot X_t + W_H^C \cdot H_{t-1} + B^C), \\ O_t &= \sigma(W_X^O \cdot X_t + W_H^O \cdot H_{t-1} + B^O), \\ C_t &= F_t \times C_{t-1} + I_t \times \widetilde{C}_t, \\ H_t &= O_t \times th(C_t), \end{aligned} \tag{A4}$$

where C_t, H_t represent the cell's slow and fast hidden states, respectively, and W_X^i, W_H^i, B^i are trained weights and biases. The operator $\times$ stands for element-wise product. The intuitive motivation behind the 'helping gates' used in LSTM cells are roughly as follows:

$\boldsymbol{F_t}$ – forget gate; determines what parts of state C should be forgotten - for 0 completely reject previous information, for 1 leave it untouched,

$\boldsymbol{I_t}$ – input gate; analogous to F_t, determines what new information should be added to the state C,

$\boldsymbol{\widetilde{C}_t}$ – candidates gate; determines possible new information to be added to C,

$\boldsymbol{O_t}$ – output gate; determines what values of the updated state C should be passed to the state H.

Interested readers can refer to Jozefowicz, Zaremba & Sutskever (2015) for an investigation of alternate designs.

Finally, we can stack multiple LSTMs to 'deepen' the network by remembering fast hidden states H_t for the whole sequence and passing it as an input to the next LSTM cell (the connection is denoted by the upward arrow in Fig. A1).

A2 ConvLSTM

In the case of temporal *and* spatially correlated data, we might require convolutional LSTM layers (Shi et al. 2015). Convolutional LSTM layers have an identical structure to the generic LSTM discussed above, with the change of weight matrices representing convolutions over spatial dimensions. We can therefore write

$$\begin{aligned} F_t &= \sigma(W_X^F * X_t + W_H^F * H_{t-1} + B^F), \\ I_t &= \sigma(W_X^I * X_t + W_H^I * H_{t-1} + B^I), \\ \widetilde{C}_t &= \mathrm{th}(W_X^C * X_t + W_H^C * H_{t-1} + B^C), \\ O_t &= \sigma(W_X^O * X_t + W_H^O * H_{t-1} + B^O). \end{aligned} \tag{A5}$$

Updates of the cell's states follow same equations as before. We note that in this research we are not using 'peephole connections' (e.g. Shi et al. 2015).

APPENDIX B: DETAILED ARCHITECTURES

Here, we present the detailed structure of all of the three NNs we use in this work. CNN, ConvRNN, and SummaryRNN architectures are summarized in Tables B1, B2, and B3, respectively. 'Layer Shape' describes a kernel size (for Conv, MaxPool and ConvLSTM layers) or a number of neurons (for LSTM and FC layers). 'Tensor Shape' denotes tensor dimensions after a particular layer. The input tensor consists of frequency, two spatial sky-plane components, and 'channels' dimensions, respectively.

Table B1. Best CNN model. The input tensor consists of frequency, two spatial sky-plane components and 'channels' dimensions, respectively.

Layer type	Layer shape	Tensor shape	Params
Input		(526, 25, 25, 1)	0
3D Conv	(8, 8, 8)	(519, 18, 18, 128)	65 664
3D MaxPool	(2, 2, 4)	(129, 9, 9, 128)	0
Batch Norm		(129, 9, 9, 128)	256
2D Conv	(4, 4, 1)	(129, 6, 6, 128)	262 272
2D MaxPool	(2, 2, 1)	(129, 3, 3, 128)	0
Batch Norm		(129, 3, 3, 128)	256
1D Conv	(1, 1, 4)	(126, 3, 3, 128)	65 664
1D MaxPool	(1, 1, 2)	(63, 3, 3, 128)	0
Batch Norm		(63, 3, 3, 128)	256
2D Conv	(3, 3, 1)	(63, 1, 1, 128)	147 584
Batch Norm		(63, 1, 1, 128)	256
1D Conv	(1, 1, 4)	(60, 1, 1, 128)	65 664
1D MaxPool	(1, 1, 2)	(30, 1, 1, 128)	0
Batch Norm		(30, 1, 1, 128)	256
1D Conv	(1, 1, 4)	(27, 1, 1, 128)	65 664
1D MaxPool	(1, 1, 2)	(13, 1, 1, 128)	0
Batch Norm		(13, 1, 1, 128)	256
1D Conv	(1, 1, 4)	(10, 1, 1, 128)	65 664
1D MaxPool	(1, 1, 2)	(5, 1, 1, 128)	0
Batch Norm	–	(5, 1, 1, 128)	256
Flatten	–	(640)	0
Dropout	–	(640)	0
FC	512	(512)	328 192
Batch Norm	–	(512)	1 024
FC	256	(256)	131 328
Batch Norm	–	(256)	512
FC	64	(64)	16 448
FC	8	(16)	1040
FC	4	(4)	68
		Total	1218 580

Table B2. Best ConvRNN model. The input tensor consists of frequency, two spatial sky-plane components, and 'channels' dimensions, respectively. 'TD' labels time distributed layer, meaning it is shared between frequencies.

Layer type	Layer shape	Tensor shape	Params
Input		(526, 25, 25, 1)	0
2D ConvLSTM	(8, 8)	(526, 18, 18, 32)	270 464
2D MaxPool (TD)	(2, 2)	(526, 9, 9, 32)	0
Batch Norm		(526, 9, 9, 32)	64
2D ConvLSTM	(4, 4)	(526, 6, 6, 64)	393 472
2D MaxPool (TD)	(2, 2)	(526, 3, 3, 64)	0
Batch Norm	–	(526, 3, 3, 64)	128
Flatten (TD)	–	(526, 576)	0
Dropout (TD)	–	(526, 1576)	0
LSTM	128	(526, 128)	361 472
Batch Norm	–	(526, 128)	256
LSTM	128	(526, 128)	132 096
Batch Norm	–	(526, 128)	256
LSTM	64	(526, 64)	49 664
Batch Norm	–	(526, 64)	128
LSTM	64	(526, 64)	33 280
Batch Norm	–	(526, 64)	128
LSTM[13]	32	(32)	12 544
Batch Norm	–	(32)	64
FC	32	(32)	1056
FC	16	(16)	528
FC	8	(8)	136
FC	4	(4)	36
		Total	1255 772

[13] Keeping only the final hidden state at the end.

Table B3. Best SummaryRNN model. The input tensor consists of frequency, two spatial sky-plane components, and 'channels' dimensions, respectively. 'TD' labels time distributed layer.

Layer type	Layer shape	Tensor shape	Params
Input		(526, 25, 25, 1)	0
2D Conv (TD)	(8, 8)	(526, 18, 18, 64)	4160
2D MaxPool (TD)	(2, 2)	(526, 9, 9, 64)	0
Batch Norm		(526, 9, 9, 64)	128
2D Conv (TD)	(4, 4)	(526, 6, 6, 128)	131 200
2D MaxPool (TD)	(2, 2)	(526, 3, 3, 128)	0
Batch Norm		(526, 3, 3, 128)	256
Flatten (TD)		(526, 1152)	0
Dropout (TD)		(526, 1152)	0
FC (TD)[14]	128	(526, 128)	147 584
Batch Norm		(526, 128)	256
LSTM	128	(526, 128)	132 096
Batch Norm		(526, 128)	256
LSTM	128	(526, 64)	49 664
Batch Norm		(526, 64)	128
LSTM	64	(526, 64)	33 280
Batch Norm		(526, 64)	128
LSTM[13]	32	(32)	12 544
Batch Norm		(32)	64
FC	32	(32)	1056
FC	16	(16)	528
FC	8	(8)	136
FC	4	(4)	36
		Total	513 500

[14]Final layer of the 2D compression, i.e. summary space.

This paper has been typeset from a TEX/LATEX file prepared by the author.