Abstract

We develop a deep-learning technique to infer the nonlinear velocity field from the dark matter density field. The deep-learning architecture we use is a “U-net” style convolutional neural network, which consists of 15 convolution layers and 2 deconvolution layers. This setup maps the three-dimensional density field of 32,000 voxels to the three-dimensional velocity or momentum fields of 2048 voxels. Through the analysis of the dark matter simulation with a resolution of 2h⁻¹ Mpc, we find that the network can predict the velocity, nonlinearity, complexity, and vorticity of the velocity and momentum fields, as well as the power spectra of their value, divergence, and vorticity and its prediction accuracy reaches the range of k ≳ 1.4 h Mpc⁻¹ with a relative error ranging from 1% to ≤10%. A simple comparison shows that neural networks may have an overwhelming advantage over perturbation theory in the reconstruction of velocity or momentum fields.

Unified Astronomy Thesaurus concepts: Large-scale structure of the universe (902); Cosmic web (330); Dark matter distribution (356); Cosmology (343); Astrostatistics (1882)

1. Introduction

The large-scale structure (LSS) of the universe is a key observational probe to study the physics of dark matter, dark energy, gravity, and cosmic neutrinos. In the next 10 years, stage IV surveys, including DESI, EUCLID, LSST, WFIRST, and CSST, will begin to map out an unprecedented large volume of the universe with extraordinary precision. It is of critical importance to have statistical tools that can reliably extract the physical information in the LSS data.

The peculiar velocities of the galaxies, sourced by the “initial” inhomogeneities, is an excellent probe for the physics of the LSS, enabling us to better study or measure such quantities as the redshift space distortions (Kaiser 1987; Jackson 1972), baryon acoustic oscillations (Eisenstein et al. 2005, 2007), the Alcock–Paczynski effect (Alcock & Paczyński 1979; Li et al. 2014, 2015, 2016; Ramanah et al. 2019), the cosmic web (Bardeen et al. 1986; Hahn et al. 2007; Forero-Romero et al. 2009; Hoffman et al. 2012; Forero-Romero et al. 2014; Fang et al. 2019), the kinematic Sunyaev–Zeldovich effect (Sunyaev & Zeldovich 1972, 1980), and the integrated Sachs–Wolfe effect (Sachs & Wolfe 1967; Rees & Sciama 1968; Crittenden & Turok 1996).

Observationally, the measurement of the peculiar velocities is a difficult task, as it requires redshift independent determination of the distance, which is usually accomplished via distance indicators such as type Ia supernovae (Phillips 1993; Riess et al. 1997; Radburn-Smith et al. 2004; Turnbull et al. 2012; Mathews et al. 2016), the Tully–Fisher relation (Tully & Fisher 1977; Masters et al. 2006, 2008), and the fundamental plane relation (Dressler et al. 1987; Djorgovski & Davis 1987; Springob et al. 2007). As an alternative approach, one can “reconstruct” the cosmic velocity field from the density field based on their relationship described by theories. Here the difficulty is the complexity caused by the nonlinear evolution of the structures. Numerous works have been done in this direction. For more details, one can check Nusser et al. (1991), Bernardeau (1992), Zaroubi et al. (1995), Croft & Gaztanaga (1997), Bernardeau et al. (1999), Kudlicki et al. (2000), Branchini et al. (2002), Mohayaee & Tully (2005), Lavaux et al. (2008), Bilicki & Chodorowski (2008), Kitaura et al. (2012), Wang et al. (2009), Jennings & Jennings (2015), and Ata et al. (2017).

Recently machine-learning algorithms, especially those based on deep neural networks, are becoming promising toolkits for the study of complex data that are difficult to solve by traditional methods. So far, this technique has been applied to almost all subfields of cosmology, including weak gravitational lensing (Schmelze et al. 2017; Gupta et al. 2018; Springer et al. 2020; Fluri et al. 2019; Jeffrey et al. 2020; Merten et al. 2019; Peel et al. 2019; Tewes et al. 2019), the cosmic microwave background (Caldeira et al. 2018; Rodríguez et al. 2018; Perraudin et al. 2019; Münchmeyer & Smith 2019; Mishra et al. 2019), the large-scale structure (Ravanbakhsh et al. 2017; Lucie-Smith et al. 2018; Modi et al. 2018; Berger & Stein 2019; He et al. 2019; Lucie-Smith et al. 2019; Pfeffer et al. 2019; Ramanah et al. 2019; Tröster et al. 2019; Zhang et al. 2019; Mao et al. 2021; Pan & Liu 2020), gravitational waves (Dreissigacker et al. 2019; Gebhardt et al. 2019), cosmic reionization (La Plante & Ntampaka 2018; Gillet et al. 2019; Hassan et al. 2019, 2020; Chardin et al. 2019), and supernovae (Lochner et al. 2016; Moss 2018; Ishida et al. 2019; Li et al. 2019; Muthukrishna et al. 2019). For more details, one can refer to Mehta et al. (2019), Jennings et al. (2019), Carleo et al. (2019), Ntampaka et al. (2019), and the references therein.
In this paper, we apply deep-learning techniques to reconstruct the velocity field from the dark matter density field. This converts the reconstruction problem to a nonlinear mapping between the two fields, which is achieved via a deep neural network with a U-net style architecture. This paper is organized as follows. In Section 2, we introduce the data set and data processing methods we use. In Section 3, we discuss our neural network, including the construction of our neural network, the selection of parameters and details of training, etc. Section 4 presents the main results, and Section 5 represents the conclusion and discussion.

2. Training and Testing Data Sets

The training and testing samples are generated using the C0moving Lagrangian Acceleration (COLA) code (Tassev et al. 2013). COLA computes the evolution of dark matter particles in a frame that is comoving with observers following trajectories predicted by the Lagrangian perturbation theory (LPT), in order to accurately deal with the small-scale structures, without sacrificing the accuracy of large scales. Being hundreds of times faster than N-body simulations, it still maintains a good accuracy from very large to highly nonlinear scales.

We generate a set of 14 simulations, assuming a ΛCDM cosmology $\Omega_m = 0.31$, $\Omega_b = 0.05$, $\sigma_8 = 0.83$, $n_s = 0.96$, and $H_0 = 67.77 \text{ km}^{-1} \text{Mpc}^{-1}$. Each of the simulations is run within a cube with a volume of $(512 \ h^{-1} \text{ Mpc})^3$ using 512$^3$ dark matter particles, having a mean separation of 1 $h^{-1} \text{ Mpc}$ per dimension. The output at $z = 0$ is then used for the main part of our analysis.

The cloud-in-cell (CIC) algorithm is adopted for constructing the density and momentum fields from the outputs. Since the momentum has three dimensions, for each sample we need to construct three fields describing $p_x$, $p_y$, and $p_z$, respectively. The division of the momentum and density fields then leads to three velocity fields, i.e., $v_x$, $v_y$, and $v_z$. For all fields, we choose a resolution of $(2 \ h^{-1} \text{ Mpc})^3$, corresponding to 256$^3$ voxels.

In practice, we further split the density and momentum/velocity voxels into smaller subcubes before feeding them to the neural network. We take such a process based on the following considerations:

1. Learning a larger cube requires a larger number of neurons or layers in the network, making the training more difficult and expensive.
2. Dealing with large fields is limited by memory constraints, especially if GPUs are used in the training process.
3. By using small cubes as training samples, we force the neural network to focus on interpreting and predicting the small-scale, nonlinear patterns in the velocity fields. The large-scale velocity field, which can be easily estimated using perturbation theory, is not our focus.

To avoid possible inaccuracy and complexity brought by the boundary effects, the neural network is designed to map the density fields into momentum fields having a smaller size. For each momentum field, we take a 240$^3$ voxel subfield from it, cut the subfield into 1728 20$^3$-voxel subcubes, and set the subcubes as the targets (i.e., outputs) of the neural network. The inputs of the network are a series of 32$^3$ voxel density fields sharing the same centers with those momentum fields. In this way, 75% voxels (lying near the outer boundary of the density fields) serve as adjacent points, for the purpose of enhancing accuracy 1.

Furthermore, since the density values span three orders of magnitude, it is difficult for the neural network to establish an accurate mapping. Thus we use the following logarithmic transform to mitigate this problem

$$\bar{p}(x, y, z) = \ln(p(x, y, z) + 1).$$

By using their log values, we greatly decrease the variance. Moreover, the distribution of large-scale structure density is close to the lognormal distribution, so we can use the above expression to convert it into an approximate normal distribution (Falck et al. 2012; Neyrinck et al. 2009; Kitaura & Angulo 2012).

The 32$^3$ voxel fields are split into training, verification, and test sets, among which the training set accounts for 60%, the verification set accounts for 30%, and the test set accounts for 10% of the total data. The single batch number for the training is set as 6.

3. Neural Network Architecture

We adopt a “U-net” style architecture, which is built upon the convolutional network and modified in a way that it has better performance in imaging analysis.

As mentioned in the previous section, the entire simulation box was divided into 32$^3$ voxel subcubes with a size of $(64 \ h^{-1} \text{ Mpc})^3$, and mapped into 20$^3$ voxel velocity fields. 64 $h^{-1} \text{ Mpc}$ is large enough to capture the nonlinear features in the field, while reducing the input data complexity, thus reducing the required number of neurons. Accordingly, the overall structure of our network is designed as follows (see Figure 2):

1. First, the input 32 voxel density fields are fed into two convolution layers, which convolve the inputs and pass the resulting feature fields to the next-level layers. To capture the abundant features in the 3D LSS, each layer has 128 filters, while each filter has a shape of 3$^3$. The latter configuration is adopted throughout our network. These two convolution layers are designed to have zero-padding and 1-stride (in what follows “same convolution”), so that their outputs have the same dimension to their inputs.
2. Then, the feature fields are convolved by 128 3$^3$-filters, but using a stride of 2. Therefore, the outputs are reduced to the size of 16$^3$. In this step we use the convolution (stride = 2) to effectively decrease the dimensions of the feature maps, and thus reduce the number of parameters to learn and the amount of computation performed in the network.
3. To further extract features and compress them, the 16$^3$ voxel feature fields are then processed by two same convolution and one convolution (stride = 2), for further feature extraction and compression. Here the three layers have as many as 256 filters, as we expect more features when entering a deeper-level regime.
4. The outputs of the previous layers, i.e., 256 8$^3$ voxel fields, are passed to two same convolution layers having 512 3$^3$ filters in each, to further extract features.

---

9 One small problem is that at some lattice points the value of the density is estimated to be zero. We assign them the background velocity, which is equal to the mean momentum divided by the mean density.
5. After that, a series of deconvolution layers are placed to conduct “inverse convolution” and achieve reconstruction. The 512 8$^3$ voxel fields are first deconvolved by 256 3$^3$ filters to produce 16$^3$ voxel fields, then convolved by 256 3$^3$ filters for further information extraction, and finally deconvolved by 128 3$^3$ filters to recover 32$^3$ voxel fields. The deconvolution is achieved via transpose convolution layers$^{10}$ with stride 1.  
6. Finally, the 32$^3$ voxel feature fields are passed to six convolution layers without padding (in what follows “valid convolution”). In each valid convolution the 3$^3$-filters decrease the size of the data by 2, so the final output has a shape of 20$^3$. They are passed to a deconvolution layer with three 3$^3$ filters and stride 1 to build up a 20$^3$ voxel cube with three-dimensional velocity as the final output.

In summary, the network is composed of a series of convolution and deconvolution layers and has a symmetric structure. It can be generally considered as an encoder network followed by a decoder network. In this way, it not only identifies features at the pixel level, but projects the features learned at different stages of the encoder onto another pixel space.

A lot of detailed designs are adopted to guarantee the performance of the network. We summarize them as follows:

1. In the decoder part, we adopted transpose convolution, instead of upsampling, as the deconvolution layer. Compared with the latter design, transpose convolution does a much better job in dealing with the nonlinearities in the fields. Based on the same consideration, in the encoder part, we also use transpose convolution, instead of max- or mean-pooling, to reduce the data.
2. After each convolution layer we place one BatchNormalization (BN) layer and one activation layer. The former one is added to prevent the overfitting of the model, reduce the training cost, and improve the training speed. The latter one, for which we use rectified linear unit (ReLU) $f(x) = \max(x, 0)$, is crucial for the neural network, since it brings nonlinearity into the system.
3. Each deconvolution is followed by a cropping layer, to match the shape of the preceding encoder convolutional density field so as to meet the concatenate condition. We crop both sides with the same pixel to guarantee each side has the same weight to the velocity field.
4. After every deconvolution, we concatenate the higher resolution feature fields from the encoder network with the deconvolved features, in order to better learn representations in the following convolutions. Since the decoder is a sparse operation, we need to fill in more details from earlier stages.
5. During the training, we randomly shuffled the input training samples of each epoch to prevent the effect of overfitting due to the similarity of adjacent fields.

### 4. Result

In the following we compare the neural network outputs with the input truth and the linear perturbation theory expectations. As mentioned in the previous subsection, in order to suppress the boundary effect in the training, the output of the neural network is a 20$^3$-voxel field, located in the center of the 32$^3$-voxel input field. Here we have already put together those subcubes in a larger field (Figure 1).

#### 4.1. Pixel-to-pixel Comparison

Figure 3 shows three slices selected from the testing samples. They all have a size 40 $h^{-1}$ Mpc $\times$ 40 $h^{-1}$ Mpc and a thickness 2 $h^{-1}$ Mpc. In all figures, we show the original “truth” velocity field, the predictions of the neural network and the linear perturbation theory, and also their residuals to the original velocity field. Plotted in the lower-left corners are the density fields based on which the velocity fields are derived.

In all cases it is clear that the neural network achieves a better performance than the linear perturbation theory:

1. The linear perturbation theory works well in the regime where the density and velocity is low (e.g., see the lower-right corner of the middle and lower panels). In the lower-right corner of the lowest panel, the performance of the perturbation theory is even better than the neural network, possibly because the latter puts most effort on predicting the nonlinear regions.
2. The linear perturbation theory completely fails in the nonlinear regions with relatively large density and velocity. But the neural network still works well in these regions.
3. The most interesting cases are those corresponding to merging situations where two regions with opposing bulk velocities collide into each other. This is shown in the lower-left part of the uppermost panel, the upper-left corner of the middle panel, and the left part of the lowest
panel. While in these regions the perturbation theory completely fails, the neural network still works well in reconstructing the velocities.

To quantify the performance of the neural network, for all slices we plot the corresponding histograms of $|v|$, $|v_{\text{residual}}|/|v_{\text{true}}|$, and $\cos \theta$, where $\theta$ is the angle between the original and the predicted velocities.

We find the neural network correctly recovers the distribution of $|v|$. However, the linear perturbation theory tends to overpredict the velocity in the dense regions. In the three slices, when checking the distribution of $|v|$, the original fields give

$$|v| = 428 \pm 188, 186 \pm 103, 432 \pm 208 \text{ km s}^{-1} \text{ (original)},$$

while the neural network predictions give

$$|v| = 376 \pm 194, 177 \pm 106, 368 \pm 193 \text{ km s}^{-1} \text{ (U-net)}.$$  

(2)

In comparison, the linear perturbation theory predictions are

$$|v| = 649 \pm 400, 386 \pm 354, 797 \pm 576 \text{ km s}^{-1} \text{ (PT).}$$

(4)

Comparing $|v_{\text{residual}}|$, the neural network results are

$$|v_{\text{residual}}| = 126 \pm 72, 65 \pm 41, 152 \pm 70 \text{ km s}^{-1} \text{ (U-net),}$$

(5)

while the linear perturbation theory yields

$$|v_{\text{residual}}| = 281 \pm 349, 298 \pm 299, 407 \pm 488 \text{ km s}^{-1} \text{ (PT).}$$

(6)

The latter results are much worse. The residual velocities of the neural network results are 3–4 times smaller than the linear perturbation theory results.

Finally, the neural network performs better than linear theory in predicting the directions of the flows. They have $|\cos \theta| = 0.93 \pm 0.19, 0.9 \pm 0.27, 0.89 \pm 0.26$ for the three slices, while in the case of linear perturbation theory the results are $0.92 \pm 0.26$, $0.49 \pm 0.69$, and $0.92 \pm 0.22$. The neural network results are closer to 1 and with a smaller standard deviation.

Similar result can be seen in Figure 4. In the nonlinear regime, linear perturbation theory completely fails, while the U-net architecture can still correctly recover the momentum. When checking the distribution of $|p|$, the original fields give

$$|p| = 2371 \pm 4613 \text{ km s}^{-1} \text{ (original),}$$

(7)

while the neural network predictions give

$$|p| = 2415 \pm 4361 \text{ km s}^{-1} \text{ (U-net).}$$

(8)

In comparison, the linear perturbation theory predictions are

$$|p| = 1030 \pm 2318 \text{ km s}^{-1} \text{ (PT).}$$

(9)

In the middle panel, we show that the neural network also performs much better in reconstructing the curl of the momentum field.

Another important quantity to characterize is the divergence of the velocity field, given its relevance to study superclusters and the cosmic web (Hoffman et al. 2012; Peñaranda-Rivera et al. 2020). So we analyze the divergence of the velocity field predicted by the neural network and compare it with the linear
Figure 3. Three slices selected from the testing sample. In regions where two bulks of matter collide and merge, the velocity is highly nonlinear. For each slice, we plot the velocity field of original input (top left), U-net prediction (top middle), and perturbation theory prediction (top right). In the bottom row, we also plot the corresponding density field (bottom left), residual of U-net prediction (bottom middle), and PT prediction (bottom right). In the right column, we plot the histograms of $v$, $|v_{\text{residual}}|/|v_{\text{true}}|$, and $|\cos \theta|$. All these suggest that the performance of neural network is much better.
Figure 4. We compare the momentum, momentum curl, and velocity divergence. Similar to Figure 3, we plot the field of original input (top left), U-net prediction (top middle), perturbation theory prediction (top right), corresponding density field (bottom left), residual of U-net prediction (bottom middle), and PT prediction (bottom right) as well as the histograms of quantities, their residuals, and angle. We find the U-net prediction is almost the same as the truth one, while the linear perturbation theory prediction loses many detailed structures. These all suggest that the performance of the neural network is much better.
perturbation theory. We find again that the neural network outperforms linear perturbation theory. In Figure 4, the divergence of velocity field predicted by the neural network is similar to the real one, while the linear perturbation theory has a larger variance.

In addition, we also made a cell-to-cell comparison of the θ_v and δ−δ_p distribution of the truth field and the U-net or PT predicted fields, in a 480 h⁻¹ Mpc box with cell-size 2 h⁻¹ Mpc. Figure 5 shows that the scattering pattern of the U-net predicted field is basically consistent with that of the truth field. In comparison, the PT method leads to a significantly wrong θ_v distribution, and also seriously overpredicts the curl value of many cells.

4.2. Power Spectrum

We now proceed to check the clustering properties of the fields. The most commonly used statistics in cosmological studies are the two-point correlation function measured in configuration space, or the power spectrum measured in Fourier space. In what follows, we compute the two-point correlation function and power spectra of specific quantities defined as

\[ \xi_{AA}(l) = \langle \delta_A(r) \delta_A(r') \rangle \]  
\[ P_{AA}(k) = \int d^3 r \xi(r) e^{i k \cdot r}, \]  

where the angle bracket represents the average of the whole sample, and A denotes the physical quantities we choose to investigate. In this analysis, the following power spectrum is taken into account,

\[ P_{\theta_v \theta_v}, P_{\theta_v \theta_p}, P_{\theta_p \theta_p}, P_{\theta_p \theta_\delta}, \]  

where \(|v| = \sqrt{v_x^2 + v_y^2 + v_z^2}, \theta_v \equiv \Delta \cdot v, p = |v| \delta\) is the momentum, and \(\Theta_p \equiv \Delta \times p\). In order to compare the difference between the reconstructed field and the actual field, we define

\[ T(k) = \frac{P_{\text{predicted}}(k)}{P_{\text{true}}(k)} \]  

Figure 6 shows the \(P_{\theta_v \theta_v}, P_{\theta_v \theta_p}, P_{\theta_p \theta_p}, P_{\theta_p \theta_\delta}\) of U-net and PT methods and their residuals to the actual power spectrum. Tables 1–4 compare the ratio of the power spectrum of PT and U-Net in different physical quantities mentioned above to the real power spectrum at \(k = 0.2, 0.6,\) and 1.0. When checking the results of \(P_{\theta_v \theta_v}\), the neural network much better recovers its value in the quasilinear regime of \(k \lesssim 0.2\ h^{-1}\) Mpc⁻¹. In particular, we find \(\lesssim 20\%\) discrepancy in \(P_{\theta_v \theta_v}\) within the range of \(0.2\ h^{-1} \lesssim k \lesssim 1.4\ h^{-1}\) Mpc⁻¹. The largest discrepancy occurs at \(k \approx 0.272\ h^{-1}\) Mpc⁻¹, corresponding to a \(T(k)\) of 0.801. In contrast, the perturbation theory result always has a discrepancy of \(T(k) \approx 1.8–5.3\).

Similar results are found when comparing the other two power spectra. In \(P_{\theta_p \theta_p}\), we find a \(\lesssim 8.2\%\) discrepancy within the range of \(0.2\ h^{-1} \lesssim k \lesssim 1.4\ h^{-1}\) Mpc⁻¹. The largest discrepancy, at \(k \approx 0.816\ h^{-1}\) Mpc⁻¹, corresponds to a \(T(k)\) of 0.918, while the perturbation theory result always has a discrepancy of \(T(k) \approx 15–47\).
The quantities $P_{\parallel\parallel}$, $P_{\theta\theta}$, $P_{pp}$, $P_{\theta_p\theta_p}$ are computed in 64 boxes with the size of $120^3 \, h^{-1} \, \text{Mpc}^3$. The gray line, blue line, and pink line represent the real field, U-net prediction field, and the linear perturbation theory field, respectively. The U-net prediction power spectrum performs much better than linear perturbation theory. The bottom figures show the deviation of the predicted power spectrum to real field. The $y$-axis $\log_{10} T$ is defined as $\log_{10} P_{\text{predicted}}/P_{\text{true}}$. All the figures within the range of $0.2 \leq k \leq 1.5$ show the U-net prediction approximates 0 while the linear perturbation theory has great deviation.

**Table 1**
Values of $P_{\parallel\parallel}/P_{\text{true}}$, Sampled at $k = 0.2$, 0.6, and 1.0

| k (hMpc$^{-1}$) | 0.2 | 0.6 | 1.0 |
|-----------------|-----|-----|-----|
| $P_{\parallel\parallel}/P_{\text{true}}$, linear perturbation theory | 2.259 | 5.245 | 3.516 |
| $P_{\parallel\parallel}/P_{\text{true}}$, U-net | 0.818 | 1.123 | 1.011 |

**Table 2**
Values of $P_{pp}/P_{\text{true}}$, Sampled at $k = 0.2$, 0.6, and 1.0

| k (hMpc$^{-1}$) | 0.2 | 0.6 | 1.0 |
|-----------------|-----|-----|-----|
| $P_{pp}/P_{\text{true}}$, linear perturbation theory | 20.860 | 44.746 | 43.479 |
| $P_{pp}/P_{\text{true}}$, U-net | 1.068 | 0.920 | 0.929 |

**Table 3**
Values of $P_{\theta\theta}/P_{\text{true}}$, Sampled at $k = 0.2$, 0.6, and 1.0

| k (hMpc$^{-1}$) | 0.2 | 0.6 | 1.0 |
|-----------------|-----|-----|-----|
| $P_{\theta\theta}/P_{\text{true}}$, linear perturbation theory | 1.584 | 7.917 | 20.731 |
| $P_{\theta\theta}/P_{\text{true}}$, U-net | 0.956 | 0.996 | 1.011 |

**Table 4**
Values of $P_{\theta_p\theta_p}/P_{\text{true}}$, Sampled at $k = 0.2$, 0.6, and 1.0

| k (hMpc$^{-1}$) | 0.2 | 0.6 | 1.0 |
|-----------------|-----|-----|-----|
| $P_{\theta_p\theta_p}/P_{\text{true}}$, linear perturbation theory | 14.806 | 24.086 | 18.065 |
| $P_{\theta_p\theta_p}/P_{\text{true}}$, U-net | 1.048 | 0.907 | 0.95 |
We find $\lesssim 18\%$ discrepancy in $P_{0,0}$ within the range of $0.2\, h\, \text{Mpc}^{-1} \leq k \leq 1.4\, h\, \text{Mpc}^{-1}$. The largest discrepancy, at $k \approx 1.355\, h\, \text{Mpc}^{-1}$, corresponds to a $T(k)$ of 0.818, while perturbation theory consistently exhibits a discrepancy of $T(k) \approx 1.3-2.2$.

There is a discrepancy of $\lesssim 9.2\%$ in $P_{6,6}$ within the range of $0.2\, h\, \text{Mpc}^{-1} \leq k \leq 1.4\, h\, \text{Mpc}^{-1}$. The largest discrepancy is seen at $k \approx 0.604\, h\, \text{Mpc}^{-1}$ with a $T(k)$ value of 0.907, while perturbation theory has a discrepancy of $T(k) \approx 11-31$.

It is worth noting here that since the outputs of our U-net only have a size of $40\, h^{-1}\, \text{Mpc}$, the spatial sampling limits our ability to accurately recover the large-scale power spectra at $k < 0.2\, h\, \text{Mpc}^{-1}$. This can be improved by making corrections on large scales, or simply increase the sizes of the input or output fields. Since the major focus of this work is to check the capability of the neural network in predicting small-scale, nonlinear velocity fields, we will not discuss this issue in detail.

5. Discussion and Conclusions

In this paper, we applied a deep-learning technique to reconstruct the velocity field from the dark matter density field, which has a resolution of $2\, h^{-1}\, \text{Mpc}$. To this end we implement a “U-net” neural network, consisting of 15 convolution layers and 2 deconvolution layers with 48,690,307 parameters. The network maps the $3^2$-voxel input density field to velocity and momentum fields having size of $20^3$, so as to avoid boundary effects.

We find that the neural network manages to reconstruct the velocity and momentum fields and even outperforms the results from linear perturbation theory. The superiority of the network is more pronounced in regions where the density is relatively large and the nonlinear processes dominate. In particular, in regions where mergers take place, linear perturbation theory completely fails, while the neural network successfully recovers the velocity structure.

By conducting pixel-to-pixel comparison between the predicted velocity fields and underlying true fields, we find that the neural network can reasonably recover the distribution of $|v|$, having discrepancy of $|v_{\text{residual}}| \lesssim 150\, \text{km s}^{-1}$, while for the perturbation theory results we find $|v_{\text{residual}}| \approx 300-400\, \text{km s}^{-1}$. The neural network also predicts well the directions of the velocities compared to the true velocities.

When analyzing the clustering properties of the fields, the neural network can well recover the amplitude and shape of $P_{v|v|}$, whose error ranges from 1% to $\lesssim 10\%$ within the range of $0.2 \lesssim k \lesssim 1.5$. Similarly, the error of $P_{pp}$ is $\lesssim 8.2\%$, $P_{0,0}$ is $\lesssim 17\%$, and $P_{6,6}$ is $\lesssim 9.2\%$ at the range of $0.2 \lesssim k \lesssim 1.4$. All these results are much better than the linear perturbation theory results.

As a proof-of-concept study, our analysis demonstrates the ability of deep neural networks to reconstruct the nonlinear velocity and momentum fields from density fields. The neural network can even handle regions of shell-crossing, which is notoriously difficult within perturbation theory approaches. At the same time, there is still much room for improvement in the accuracy of the neural network, via further optimizing the architecture, enlarging the number of the training samples, or adding follow-up neural networks to fit the residuals and perform corrections.

The reconstructed peculiar velocity fields can be used for a number of studies, such as BAO reconstructions, RSD analyses, kinematic Sunyaev–Zeldovich (kSZ), supercluster analysis and the cosmic web construction. We will continue to work on this direction so that the machine-learning technique can be reliably applied to real observational data and help us uncover more of the mysteries of the universe.

We thank Kwan-Chuen Chan, Yin Li, Jie Wang, Le Zhang, and Yi Zheng for helpful discussions. This work is supported by the National SKA Program of China No. 2020SKA0110401. X.D.L. acknowledges the support from the NSFC grant (No. 11830394) and the Science and Technology Program of Guangzhou, China (No. 202002030360). C.G.S. acknowledges financial support from the National Research Foundation of Korea (NRF; #2020R1I1A1A01073494). J.F.-R. acknowledges support from COLCIENCIAS Contract No. 287-2016, Project 1204-712-50459. Y.W. is supported by NSFC grant No.11803095 and NSFC grant No.11733010. We acknowledge the use of Tianhe-2 supercomputer. We also acknowledge the use of the Kunlun cluster, a supercomputer owned by the School of Physics and Astronomy, Sun Yat-Sen University.

**References**

Alcock, C., & Paczyński, B. 1979, Natur, 281, 358

Ata, M., Kaitaara, F.-S., Chuang, C.-H., et al. 2017, MNRAS, 467, 3993

Bardeen, J. M., Bond, J. R., Kaiser, N., & Szalay, A. S. 1986, ApJ, 304, 15

Bernardeau, F. 1992, ApJL, 390, L61

Bernardeau, F., Chodorowski, M. J., Lokas, E. L., Stompor, R., & Kudlicki, A. 1999, MNRAS, 309, 543

Bilicki, M., & Chodorowski, M. J. 2008, MNRAS, 391, 1796

Branchini, E., Eldar, A., & Nusser, A. 2002, MNRAS, 335, 53

Caldeira, I., Wu, W. L. K., Nord, B., et al. 2018, A&C, 28, 100307

Carleo, G., Cirac, I., Cranmer, K., et al. 2019, RvMP, 91, 045002

Chardin, J., Uhrich, G., Aubert, D., et al. 2019, MNRAS, 490, 1055

Crittenden, R. G., & Turok, N. 1996, PhRvL, 76, 575

Croft, R. A. C., & Gaztanaga, E. 1997, MNRAS, 285, 793

Djorgovski, S., & Davis, M. 1987, ApJ, 313, 59

Dreissigacker, C., Sharma, R., Messenger, C., Zhao, R., & Prix, R. 2019, PhRvD, 100, 044009

Dressler, A., Lynden-Bell, D., Burstein, D., et al. 1987, ApJ, 313, 42

Eisenstein, D. J., Seo, H.-J., Sirko, E., & Spergel, D. N. 2007, ApJ, 664, 675

Eisenstein, D. J., Zehavi, I., Hogg, D. W., et al. 2005, ApJ, 633, 560

Falck, B. L., Neyrinck, M. C., Aragon-Calvo, M. A., Lavaux, G., & Szalay, A. S. 2012, ApJ, 745, 17

Fang, F., Forero-Romero, J., Rossi, G., Li, X.-D., & Feng, L.-L. 2019, MNRAS, 485, 5276

Fluri, J., Kacprzak, T., Lucchi, A., et al. 2019, PhRvD, 100, 063514

Forero-Romero, J., Hoffmann, Y., Gottlöber, S., Klypin, A., & Yepes, G. 2009, MNRAS, 396, 1815

Forero-Romero, J. E., Contreras, S., & Padilla, N. 2014, MNRAS, 443, 1090

Gehard, T. D., Kilbertus, N., Harry, I., & Schölkopf, B. 2019, PhRvD, 100, 063015

Gillet, N., Mesinger, A., Greig, B., Liu, A., & Ucci, G. 2019, MNRAS, 484, 282

Gupta, A., Matilla, J. M. Z., Hsu, D., & Haiman, Z. 2018, PhRvD, 97, 103515

Hahn, O., Porciani, C., Carollo, C. M., & Dekel, A. 2007, MNRAS, 375, 489

Hassan, S., Andrianomena, S., & Doughty, C. 2020, MNRAS, 494, 5761

Hassan, S., Liu, A., Kohn, S., & La Plante, P. 2019, MNRAS, 483, 2524

He, S., Li, Y., Feng, Y., et al. 2019, PNAS, 116, 13825

Hoffman, Y., Metuki, O., Yepes, G., et al. 2012, MNRAS, 425, 2049

Ishida, E. E. O., Bekk, R., Gonzalez-Gaitán, S., et al. 2019, MNRAS, 483, 2

Jackson, J. 1972, MNRAS, 156, 1P

Jeffrey, N., Lanusse, F., Lahav, O., & Starck, J.-L. 2020, MNRAS, 492, 5023
