Locality constrained autoregressive networks for lattice field theory simulations

February 22, 2023

1 The autoregressive relation and scalar lattice field theory

The systems of interest consist of a box lattice of length L in d dimensions where every position is labeled using a d-dimensional vector $\boldsymbol{x} \in [1, L]^d$. The system state/configuration is described using scalar values at every position $\phi(\boldsymbol{x})$. The configurations obey the Boltzmann distribution:

$$p\left(\{\phi(\boldsymbol{x})\}_{\boldsymbol{x}\in[1,L]^d}\right) = e^{-S[\phi]}/Z \tag{1}$$

where the *action* $S[\phi]$ is a functional of the field values $\phi(\boldsymbol{x})$. The d-dimensional positions $\boldsymbol{x}$ maybe replaced with a 1 dimensional ordering:

$$k = \left(\sum_{i=1}^{d}(x_i - 1)L^{i-1}\right) + 1 \tag{2}$$

where x_i are the components of $\boldsymbol{x}$ and $k \in [1, N = L^d]$. Based on this ordering, we can write down the probability distribution in 1 as a product of conditional distributions at every position:

$$\begin{aligned} p(\{\phi_k\}) = p(\phi_1, \phi_2 \ldots \phi_N) &= p(\phi_1)p(\phi_2|\phi_1)\ldots p(\phi_N|\phi_{N-1}\ldots\phi_2, \phi_1) \\ &= \prod_{k\in[1,N]} p(\phi_k|\phi_{<k}) \end{aligned} \tag{3}$$

This is the chain rule of conditional probabilities based on Bayes theorem or *autoregressive relation*. This mathematical relation is the basis of image and audio generation algorithms in deep learning such as MADE[4] and PixelCNN[8]. Our system of interest is the scalar lattice field theory whose action is given by:

$$S[\phi] = \sum_{\boldsymbol{x}\in[1,L]^d}\left[\phi(\boldsymbol{x})\sum_{\boldsymbol{y}}\Box(\boldsymbol{x},\boldsymbol{y})\phi(\boldsymbol{y}) + m^2\phi(\boldsymbol{x})^2 + \lambda\phi(\boldsymbol{x})^4\right] \tag{4}$$

where a, m, λ are the lattice spacing, mass and coupling respectively. Assuming open boundary conditions, we can expand the d'Alembertian term in the RHS as:

$$\sum_{\boldsymbol{x}\in[1,L]^d} \phi(\boldsymbol{x}) \sum_{\boldsymbol{y}} \Box(\boldsymbol{x},\boldsymbol{y})\phi(\boldsymbol{y}) = \sum_{\mu=1}^{d} \sum_{x_{\nu=\mu}\in[2,L-1], x_\nu\neq[1,L]} 2\phi(\boldsymbol{x})^2 - \phi(\boldsymbol{x})\phi(\boldsymbol{x}-\hat{\mu}) - \phi(\boldsymbol{x})\phi(\boldsymbol{x}+\hat{\mu})$$

and $\phi(\boldsymbol{x})$ can take any real value. Note that $S[\phi]$ contains only nearest neighbour product/interaction terms $\phi(\boldsymbol{x})\phi(\boldsymbol{x}-\hat{\mu})$ and $\phi(\boldsymbol{x})\phi(\boldsymbol{x}+\hat{\mu})$, other than powers of $\phi(\boldsymbol{x})$. This property of the action is known as *locality* which is obeyed by more complex lattice field theory systems as well[1].

2 Smaller dependency sets of conditional distributions due to nearest neighbour interactions

Examining the kth conditional probability $p(\phi_k|\phi_{<k})$ in 3, its distribution in general depends on $k-1$ values in $\phi_{<k} = \{\phi_{k-1}, \ldots \phi_1\}$. This means the complexity of these distributions can explode if the number of lattice points N is large, which is typically the case of interest. That's the reason deep neural networks have been utilized to model them for image/audio generation. However for systems with nearest neighbour interactions, the *dependency set* is significantly smaller (from my master's thesis [7]). It's easier to show this (without loss of generality) for the nearest neighbour Ising model whose action is given by:

$$S[\phi] = -\beta J \sum_{\mu=1}^{d} \sum_{x_{\nu=\mu}\in[2,L], x_{\nu\neq\mu}\in[1,L]} \phi(\boldsymbol{x}-\hat{\mu})\phi(\boldsymbol{x}) \tag{5}$$

where $\phi(\boldsymbol{x})$ takes values ± 1. Restating the autoregressive relation for the Ising model[2]:

$$\prod_{k=1}^{N} p(\phi_k|\phi_{<k}) = p(\phi) = \exp\left(-\beta J \sum_{\mu=1}^{d} \sum_{k} \phi_k \phi_{k-\hat{\mu}}\right) / Z \tag{6}$$

From Bayes theorem, we can relate this conditional probability to the unconditional probabilities of the first k and $k-1$ spins, which can in turn be written as reduced forms of the Boltzmann distribution:

$$p(\phi_k|\phi_{<k}) = \frac{p(\phi_1, \ldots \phi_k)}{p(\phi_1, \ldots \phi_{k-1})} = \frac{\sum_{\phi_{k+1}\ldots\phi_N} p(\phi)}{\sum_{\phi_k\ldots\phi_N} p(\phi)}$$

[1] In more famous words, "there's no spooky action at a distance".

[2] $k-\hat{\mu}$ should be understood as the lattice position $\boldsymbol{x}-\hat{\mu}$ where $\boldsymbol{x}$ maps to k according to the given ordering

Expanding the $p(\phi)$ for the Ising model:

$$p(\phi_k|\phi_{<k}) = \frac{\displaystyle\sum_{\phi_N,\dots\phi_{k+1}} \exp\left(-\beta J \sum_{l=k}^{N}\left(\phi_l \sum_{\mu} \phi_{l-\hat{\mu}}\right) + \delta(\phi_{<k})\right)}{\displaystyle\sum_{\phi_N,\dots\phi_k} \exp\left(-\beta J \sum_{l=k}^{N}\left(\phi_l \sum_{\mu} \phi_{l-\hat{\mu}}\right) + \delta(\phi_{<k})\right)}$$

Since the values in $\phi_{<k}$ are fixed and not summed over, the terms $\delta(\phi_{<k})$ containing only them cancel from the numerator and denominator, leaving us with:

$$p(\phi_k|\phi_{<k}) = \frac{\displaystyle\sum_{\phi_N,\dots\phi_{k+1}} \exp\left(-\beta J \sum_{l=k}^{N}\left(\phi_l \sum_{\mu} \phi_{l-\hat{\mu}}\right)\right)}{\displaystyle\sum_{\phi_N,\dots\phi_k} \exp\left(-\beta J \sum_{l=k}^{N}\left(\phi_l \sum_{\mu} \phi_{l-\hat{\mu}}\right)\right)} \tag{7}$$

Even though $\phi_{<k}$ contains $k-1$ values, the conditional probability $p(\phi_k|\phi_{<k})$ depends only on those positions within $\phi_{<k}$ that are neighbours of the positions in $\phi_{\geq k}$. We can draw the same conclusion for scalar lattice field theory by replacing the sums with integrals and including terms like ϕ_l^2 and ϕ_l^4 in the above expression. The number of elements in the dependency set is bounded above by L^{d-1} or N/L for our choice of ordering (see figure 1 for an illustration on a 2D lattice) which is an "order of magnitude" smaller than the original upper bound N. In fact, we can join 2 strips of black spins in figure 1 into a single 1D line, and the conditional distribution on ϕ_k simply depends on the values along this line.

Dependency surfaces for single and joint conditional distributions We can also mark the dependency set of $\phi_{k(\boldsymbol{x})}$ as a $d-1$ dimensional surface constructed parametrically using[3]:

$$B_{\boldsymbol{x}}(y_1, \dots y_{d-1}) = \begin{cases} [y_1, \dots, y_{d-1}, x_d] & \text{if } k(y_1, \dots, y_{d-1}, x_d) < k(\boldsymbol{x}) \\ [y_1, \dots, y_{d-1}, x_d - 1] & \text{if } k(y_1, \dots, y_{d-1}, x_d) > k(\boldsymbol{x}) \end{cases} \tag{8}$$

We'll call this the *dependency surface* at $\boldsymbol{x}$. From 7, we can write down the joint conditional probabilities for more than one variable. For example, we can

[3]The reader is urged to spend some time grasping this, perhaps with aid from figure **??** for the case of $d = 2$.

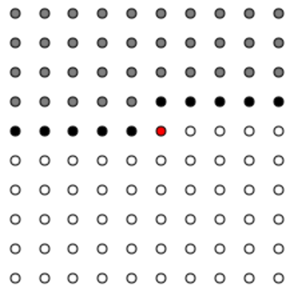

Figure 1: In the 2 dimensional 10×10 lattice above, the conditional probability $p(\phi_k|\phi_{<k})$ of the red spin depends only on the nearest neighbours of the spins in $\phi_{>k}$ (coloured white), within $\phi_{<k}$. Hence the dependency set is only the $L = 10$ spins coloured black and doesn't contain the grey ones above it.

write:

$$p(\phi_k, \phi_{k+1}|\phi_{<k}) = p(\phi_{k+1}|\phi_{<k+1})p(\phi_k|\phi_{<k})$$
$$= \frac{\sum\limits_{\phi_N,\dots\phi_{k+2}} \exp\left(-\beta J \sum\limits_{l=k}^{N}\left(\phi_l \sum\limits_{\mu}\phi_{l-\hat{\mu}}\right)\right)}{\sum\limits_{\phi_N,\dots\phi_k} \exp\left(-\beta J \sum\limits_{l=k}^{N}\left(\phi_l \sum\limits_{\mu}\phi_{l-\hat{\mu}}\right)\right)} \tag{9}$$

Here, if the x_d is the same for both k and $k+1$, then the dependency surface for this joint conditional porbability is still the same $B_{\boldsymbol{x}}$! For a particular choice of $\boldsymbol{x} = [1, \dots, 1, t]$, the dependency surface $B_{\boldsymbol{x}}$ =$[y_1, \dots, y_{d-1}, t-1]$ since the case 1 in 8 never arises. By extension of the logic in 9, the dependency of the surface $x_d = t$ can be inferred from:

$$p(\phi_{k(\boldsymbol{x})}, \dots \phi_{k(\boldsymbol{x})+L^{d-1}-1}|\phi_{<k}) = \frac{\sum\limits_{\phi_N,\dots\phi_{k+L^{d-1}-1}} \exp\left(-\beta J \sum\limits_{l=k}^{N}\left(\phi_l \sum\limits_{\mu}\phi_{l-\hat{\mu}}\right)\right)}{\sum\limits_{\phi_N,\dots\phi_k} \exp\left(-\beta J \sum\limits_{l=k}^{N}\left(\phi_l \sum\limits_{\mu}\phi_{l-\hat{\mu}}\right)\right)} \tag{10}$$

and that's simply $B_{\boldsymbol{x}}$. In simpler terms, the joint probability of the surface $x_d = t$ is conditioned only on the surface $x_d = t - 1$: it's similar to propagating a stochastic differential equation from an initial value!

3 Neural network ansatz for autoregressive sampling

We can model the distribution $p(\phi_k|B_k)$ using a neural network ansatz and sample lattice values sequentially, similar to MADE or PixelCNN. For example, we can let the outputs of the k^{th} neural network parameterize a mixture of M Gaussians:

$$\{w_j, \mu_j, \sigma_j\}_{j=1}^{M} = NN_k(B_k)$$
$$p(\phi_k|B_k) \approx \sum_j w_j \mathcal{N}(\mu_j, \sigma_j)$$

which is flexible, as well as easy to sample from. We can exploit the translational invariance of the system, drop the k subscript and sample using the same neural network NN at every position- an approximation that gets better as L gets large. This ensures the number of neural network weights do not scale with system size and also enables a scalable model where a network trained on smaller L can be reused to sample a larger lattice- which should be crucial for lattice field theories where the cost of simulating systems typically scale with the system size. Since the translational invariance is only approximate, we should expect errors to increase when we do an extrapolation to large L. It will be interesting to see if we can engineer the model architecture or the input data to address such sources of error. The log likelihood at every position can be accumulated and optimized using the REINFORCE estimator of the KL divergence between the ansatz and the unnormalized Boltzmann distribution (see [9] for a treatment of the Ising model).

We can model the 2-variable conditional distribution $p(\phi_{k+1}, \phi_k|B_k)$ from 9 using a flow-based network like RealNVP[3] where the prior distribution would be Gaussians parameterized by a convolutional network acting on B_k. It uses a much more flexible ansatz compared to mixture of Gaussians and *reparameterizable* sampling of the conditionals allows us to optimize the KL divergence directly, and mitigates issues like variance when using the REINFORCE estimator. This can essentially be a compact (and scalable if it's the 2 variable model) versions of a model that uses a flow-based network to sample the entire lattice like in [1][4]. We can also model the quantity $p\left(\{\phi_{\boldsymbol{x}}\}_{x_d=t}|\{\phi_{\boldsymbol{x}}\}_{x_d=t-1}\right)$ from 10 whose dependency set would be the surface $x_d = t-1$, using flow based networks. Since this samples one time step at a time, this model can be considered an unsupervised version of neural SDEs.

While approaches using autoregressive networks for sampling lattice field theories already exist [6], expoiting the $d-1$ dimensional dependency set means the neural network NN can be a $d-1$ dimensional convolutional network- which can be designed to model stronger dependence on positions closer to ϕ_k

[4]This is an oversimplified picture and there are differences like periodic vs open boundary conditions. Assumption of translational invariance on a finite lattice can contribute to errors which can require more careful model construction to address.

than farther ones. A notion of locality has also been incorporated in flow-based models that directly sample the whole lattice by checkerboard decomposition of the prior samples and using convolutions in coupling layers according to [5]. Our proposal of using $d-1$ dimensional dependency surface however enforces a stronger and explicit constraint directly based on locality of the action.

A fascinating outcome of lower dimensional inputs is that in the practically important case of $d=4$, it's sufficient to use 3D convolutional layers that have optimized GPU implementations in the CUDA stack or popular deep learning frameworks like PyTorch or Tensorflow- the same are typically absent for 4D convolutions. More generally, smaller input space of the proposed class of models help tackle the curse of dimensionality that usually plagues neural networks- and can be a "locality" addition to the Geometric Deep Learning framework[2].

References

[1] Michael S Albergo, Gurtej Kanwar, and Phiala E Shanahan. "Flow-based generative models for Markov chain Monte Carlo in lattice field theory". In: *Physical Review D* 100.3 (2019), p. 034515.

[2] Michael M Bronstein et al. "Geometric deep learning: going beyond euclidean data". In: *IEEE Signal Processing Magazine* 34.4 (2017), pp. 18–42.

[3] Laurent Dinh, Jascha Sohl-Dickstein, and Samy Bengio. "Density estimation using real nvp". In: *arXiv preprint arXiv:1605.08803* (2016).

[4] Mathieu Germain et al. "Made: Masked autoencoder for distribution estimation". In: *International conference on machine learning*. PMLR. 2015, pp. 881–889.

[5] Gurtej Kanwar. "Machine Learning and Variational Algorithms for Lattice Field Theory". In: *arXiv preprint arXiv:2106.01975* (2021).

[6] Di Luo et al. "Gauge invariant autoregressive neural networks for quantum lattice models". In: *arXiv preprint arXiv:2101.07243* (2021).

[7] Dinesh PR. *Analysis of Ising model using neural networks*. July 2021. URL: `http://dr.iiserpune.ac.in:8080/xmlui/handle/123456789/6014`.

[8] Aaron Van den Oord et al. "Conditional image generation with pixelcnn decoders". In: *Advances in neural information processing systems* 29 (2016).

[9] Dian Wu, Riccardo Rossi, and Giuseppe Carleo. "Unbiased Monte Carlo cluster updates with autoregressive neural networks". In: *Physical Review Research* 3.4 (2021), p. L042024.