Google Research Scalable variational Monte Carlo with graph neural ansatz[1]

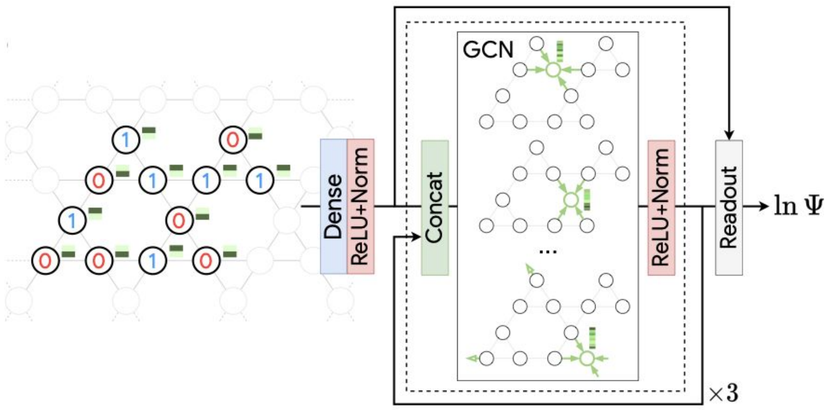
Li Yang*, Wenjun Hu**, Li Li*

*Google Research, **Department of Physics and Astronomy, University of Tennessee

Introduction

Deep neural networks have been shown as a potentially powerful ansatz in variational Monte Carlo for solving quantum many-body problems. We propose two improvements in this direction. The first is graph neural ansatz (GNA), which is a variational wavefunction universal to arbitrary geometry. GNA results in accurate ground-state energies on 2D Kagome lattices, triangular lattices, and randomly connected graphs. Secondly, we design a distributed workflow on multiple accelerators to scale up the computation. We compute Kagome lattices with sizes up to 12 x 12 x 3 sites on 128 TPU cores. The parameter sharing nature of the GNA also leads to transferability across different system sizes and geometries.

Graph Neural Ansatz



Graph neural networks can contain the inductive bias of capturing arbitrary geometries. We use a special kind of GNN, graph convolutional network, as the GNA:

- At each layer, nodes are embedded into $H^{(l)} \in \mathbb{R}^{N_{\text{site}} \times F}$
- The next layer embeddings are calculated by:

$$H^{(l+1)} = f([D^{-\frac{1}{2}} A D^{-\frac{1}{2}} H^{(l)}; H^{(l)}]W + b),$$

  where ***F*** the embedding size, ***A*** the adjacency matrix, ***D*** a node degree diagonal matrix, ***W*** and ***b*** the weights and bias, [;] denotes concatenation along feature dimension, ***f*** the nonlinear transformation (ReLU).
- We use an recurrent structure: ***W*** and ***b*** are shared across layers.

Importance Sampling Optimization (ISGO)[2]

In VMC, Markov-chain must be long enough to provide a reliable states distributions, but it is essentially an sequential process, which may not coordinate perfectly with modern deep learning software and hardware. We developed an algorithm to reuse samples for VMC optimization using importance sampling.

The original VMC updating formula is:

$$\partial_w E_v \approx \sum_x I^0 E_x^0 \partial_w \ln \Psi_x^0 - \sum_x I^0 E_x^0 \sum_x I^0 \partial_w \ln \Psi_x^0,$$

To reuse samples, an importance sampling factor is added:

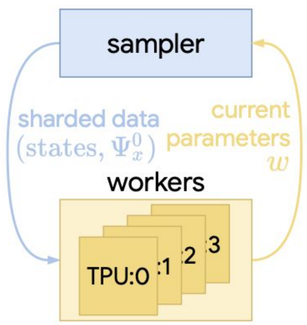
$$\partial_w E_v \approx \sum_x I_x E_x \partial_w \ln \Psi_x - \sum_x I_x E_x \sum_x I_x \partial_w \ln \Psi_x,$$

$$I_x / I^0 = \mathcal{C} |\Psi_x|^2 / |\Psi_x^0|^2, \quad \sum_x I_x / I^0 = 1$$

In practice, we use MCMC with length 5e4 and after a few iterations (~10), the model already converges good.

Scaling up ISGO to multiple accelerators

By using the efficient all-reduce sum supported by many deep learning frameworks and hardware, e.g. psum in JAX[3], ISGO can be scaled up to multiple accelerators.



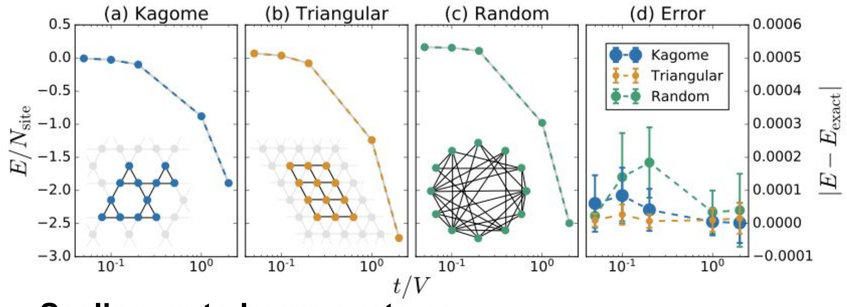
Numerical Experiments

We apply the GNA to Hard-core boson systems on various lattice geometries including 2D Kagome lattice, 2D Triangular lattice, and randomly connected graph. The hamiltonian is

$$H = -t \sum_{\langle ij \rangle} (b_i^\dagger b_j + b_j^\dagger b_i) + V \sum_{\langle ij \rangle} n_i n_j$$

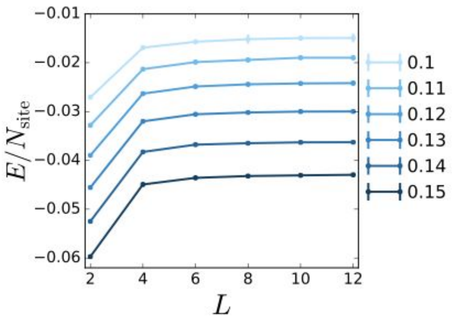
The off-diagonal elements of the hamiltonian is non-positive, so the ground state wavefunction is always positive semi-definite.

Benchmark with Exact Diagonalization



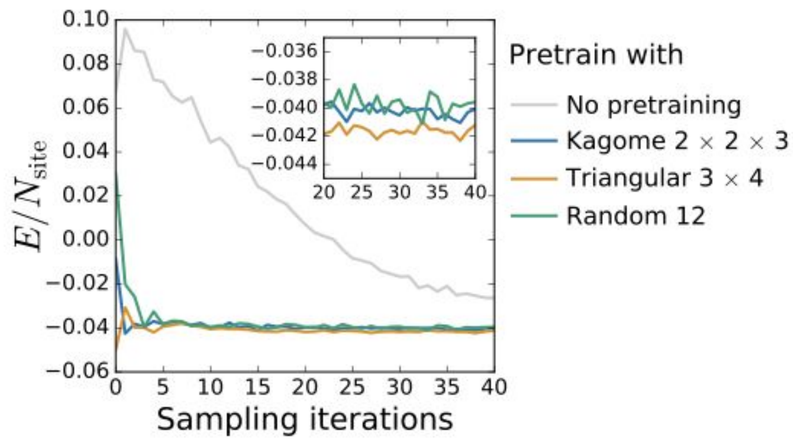
Scaling up to larger systems

For ground state energies of Kagome lattice of size L x L x 3, we use 8 TPU cores for L < 8 and up to 128 TPU cores for L up to 12. Different curves denote various t/V.



Finetuning

A GNA model can be used for arbitrary geometries without changing model architectures, which allows to pretrain and finetune.



References

[1] L. Yang, W. J. Hu and L. Li, arxiv:2011.12453

[2] L. Yang, Z. Q. Leng, G. Yu, A. Patel, W. J. Hu, and H. Pu, Phys. Rev. Research 2, 012039(R) (2020).

[3] J. Bradbury, R. Frostig, P. Hawkins, M. J. Johnson, C. Leary, D. Maclaurin, and S. Wanderman-Milne, http://github.com/google/jax.