Developing of a Transformed based architecture to solve the Time Independent Many Electron Schrodinger Equation

Table of Contents

1. Abstract
2. Introduction
3. Objectives
4. Theoretical Framework
    1. The problem
        1. The Schrodinger Equation
        2. The many electron Schrodinger Equation
    2. Approximating a solution
        1. RayLeight Quotient
    3. Using Deep Learning
        1. Fermi Net
        2. Transformers
        3. Psi Former
        4. Loss function
        5. Optimizer
        6. Flow of the architecture
5. Methodology
    1. Environment
    2. Training

---

observations 1

Abstract

With accurate solutions to the many electron Schrodinger equation all the chemistry could derived from first principles. Try to find analytical is prohibitively hard due the intrinsic relations between each component on a molecule. In this work I develop the use of a architecture based on the Transformer architecture to tackle this problem.

Introduction

The success of deep Learning across different fields like protein folding Jumper et al. (2021), visual modeling Dosovitskiy et al. (2021), ODEs solvers Raissi et al. (2019) has sparked great interest from the scientific community to apply DL methods to their fields.

Quantum Chemistry, specifically finding a good aproximmation for the Quantum Many-Body wave eqaution the is one of those places where have shown that deep learning could overpass traditional methods Luo & Clark (2019) , Qiao et al. (2020), but there is still many challenges specifically, the computational power needed for large molecules becomes prohibitively expensive.

Tackling that problem the Transformer architecture had demonstrate that scaling laws are not so much complicated for him. Cite

Motivated for that in this work I develop a transformer architecture called Psifomer. von Glehn et al. (2023)

Objectives

- Obtain a model which is able to replicate the energy ground states of certain atoms.
- Compare our model with another State of the art methods to solve the Many electrons Schrodinger equation

Overview

I provide an outline of the model architecture and procedure
In theoretical frame work we will and in methodology we will

Theoretical Framework

In order to solve the problem we have to grasp the physics laws that our solution have to follow,

The problem

We consider the follow:

The Schrodinger Equation

The schrodinger equation was presented in a series of publication made it by Schrodinger in the year 1916.

It was received pretty well by the scientific comunnity.

And its relevance is high, in principle it is able to explain all the atomic phenomena and all the facts of chemical bindings.

The many electron Schrodinger Equation

In quantum chemistry is regular used atomic units, the unit of distance is the Bohr Radious and the unit of energy is Hartree (Ha).

In its time-independent form the Schrodinger equation can be written as a eigenfunction equation.

$$\hat{H}\psi(\mathbf{x}_0, \ldots, \mathbf{x}_n) = E\psi(\mathbf{x}_1, \ldots, \mathbf{x}_n)$$

Where $\hat{H}$ is a Hermitian linear operator called the Hamiltonian and the scalar eigenvalue E corresponds to the energy of a particular solution.

$$U = \frac{1}{4\pi\varepsilon_0}\frac{e^2}{|r_i - r_j|}$$

Using atomic units we Atomic Units:

The Hamiltonian using the Quantum Chemistry units becomes:

$$\hat{H} = -\frac{1}{2}\sum\nabla^2 + \sum\frac{1}{|r_i - r_j|} - \sum\frac{Z_I}{|r_i - R_I|} + \sum\frac{Z_I Z_J}{|R_i - R_j|}$$

Where Z_I are the atomic number r_i is the distance from a reference frame

Now the Fermi Dirac Statistics tell us that this solution of this equation should be **anti symmetric** this is:

$$\psi(\ldots, \mathbf{x}_i, \ldots, \mathbf{x}_j, \ldots) = -\psi(\ldots, \mathbf{x}_j, \ldots, \mathbf{x}_i, \ldots)$$

The potential energy becomes infinite when two electrons overlap , this could be formalized via the Kato Cusp Conditions, a Jastrow factor $\exp(\mathcal{J})$. The explicit form of $\mathcal{J}$ depends on the author.

Approximating a solution

Find possible solution in the traditional way is prohibitively hard. So what people have doing and it seem that it becomes a success is guess that solution and using another techniques to improve the solution, to this guess solution we called **Ansatz**.

Once that you have your Ansatz, which normally depends on depends on certain parameters.

Variational Monte Carlo

Once that you guess an **Ansatz** you optimize using the **rayleight quotient**.

$$\mathcal{L} = \frac{\langle\psi|\hat{H}|\psi\rangle}{\langle\psi|\psi\rangle} = \frac{\int d\mathbf{r}\psi^*(\mathbf{r})\hat{H}\psi(\mathbf{r})}{\int d\mathbf{r}\psi^*(\mathbf{r})\psi(\mathbf{r})}$$

So how we optimized this. Here appears Variational Quantum Monte Carlo.

Which can be re-written as:

$$E_L(x) = \Psi_\theta^{-1}(x)\hat{H}\Psi_\theta(x)$$

$$\mathcal{L}_\theta = \mathbb{E}_{x\sim\Psi_\theta^2}[E_L(x)]$$

And here we use Metropolis algorithm to work in real life.

Using Deep Learning

They are a quite example of it.

examples Shang et al. (2025) Related work

Neural Networksee

RNN

Fermi Net

A very important work for us is: Fermi Net Pfau et al. (2020) it uses different MLP to learn the forms of the orbitals. Their ansatz is: Fermi Net

$$\psi(\mathbf{x}_i, \ldots, \mathbf{x}_n) = \sum_k \omega_k \det[\Phi^k]$$

With:

$$\begin{vmatrix} \phi_1^k(\mathbf{x}_1) & \dots & \phi_1^k(\mathbf{x}_n) \\ \vdots & & \vdots \\ \phi_n^k(\mathbf{x}_1) & \dots & \phi_n^k(\mathbf{x}_n) \end{vmatrix} = \det[\phi_i^k(\mathbf{x}_j)] = \det[\boldsymbol{\Phi}^k]$$

The elements of the determinant are obtained via

$\alpha \in \{\uparrow, \downarrow\}$

$$\mathbf{h}_i^{\ell\alpha} \leftarrow \text{concatenate}(\mathbf{r}_i^\alpha - \mathbf{R}_I, |\mathbf{r}_i^\alpha - \mathbf{R}_I| \; \forall \; I)$$

$$\mathbf{h}_{ij}^{\ell\alpha\beta} \leftarrow \text{concatenate}(\mathbf{r}_i^\alpha - \mathbf{r}_j^\beta, |\mathbf{r}_i^\alpha - \mathbf{r}_j^\beta| \; \forall \; j, \beta)$$

$$\begin{aligned} &\left(\mathbf{h}_i^{\ell\alpha}, \frac{1}{n^\uparrow}\sum_{j=1}^{n^\uparrow}\mathbf{h}_j^{\ell\uparrow}, \frac{1}{n^\downarrow}\sum_{j=1}^{n^\downarrow}\mathbf{h}_j^{\ell\downarrow}, \frac{1}{n^\uparrow}\sum_{j=1}^{n^\uparrow}\mathbf{h}_{ij}^{\ell\alpha\uparrow}, \frac{1}{n^\downarrow}\sum_{j=1}^{n^\downarrow}\mathbf{h}_{ij}^{\ell\alpha\downarrow}\right) \\ &= \left(\mathbf{h}_i^{\ell\alpha}, \mathbf{g}^{\ell\uparrow}, \mathbf{g}^{\ell\downarrow}, \mathbf{g}_i^{\ell\alpha\uparrow}, \mathbf{g}_i^{\ell\alpha\downarrow}\right) = \mathbf{f}_i^{\ell\alpha}, \end{aligned}$$

$$\begin{aligned} \mathbf{h}_i^{\ell+1\alpha} &= \tanh\left(\mathbf{V}^\ell \mathbf{f}_i^{\ell\alpha} + \mathbf{b}^\ell\right) + \mathbf{h}_i^{\ell\alpha} \\ \mathbf{h}_{ij}^{\ell+1\alpha\beta} &= \tanh\left(\mathbf{W}^\ell \mathbf{h}_{ij}^{\ell\alpha\beta} + \mathbf{c}^\ell\right) + \mathbf{h}_{ij}^{\ell\alpha\beta} \end{aligned}$$

$$\begin{aligned} \phi_i^{k\alpha}(\mathbf{r}_j^\alpha; \{\mathbf{r}_{/j}^\alpha\}; \{\mathbf{r}^{\bar{\alpha}}\}) = &\left(\mathbf{w}_i^{k\alpha} \cdot \mathbf{h}_j^{L\alpha} + g_i^{k\alpha}\right) \\ &\sum_m \pi_{im}^{k\alpha} \text{exp}\left(-|\boldsymbol{\Sigma}_{im}^{k\alpha}(\mathbf{r}_j^\alpha - \mathbf{R}_m)|\right), \end{aligned}$$

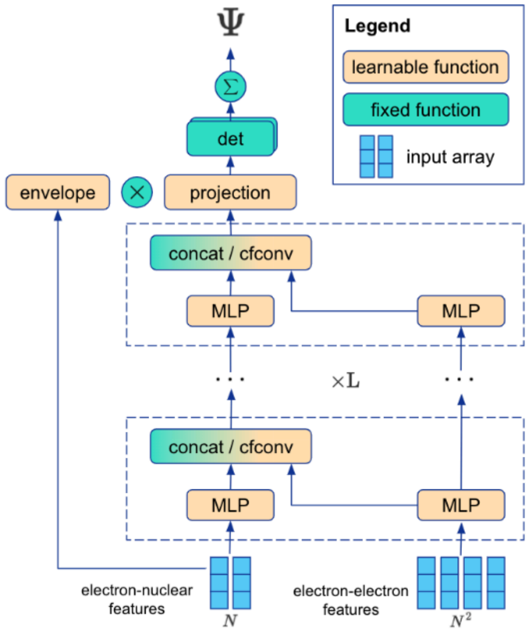
$$\phi_i^{k\alpha}(\mathbf{r}_j^\alpha; \{\mathbf{r}_{/j}^\alpha\}; \{\mathbf{r}^{\bar{\alpha}}\}) = (\mathbf{w}_i^{k\alpha} \cdot \mathbf{h}_j^{L\alpha} + g_i^{k\alpha}) \sum_m \pi_{im}^{k\alpha} \exp\left(-\left|\Sigma_{im}^{k\alpha}(\mathbf{r}_j^\alpha - \mathbf{R}_m)\right|\right)$$

.

$$\begin{aligned} \psi(\mathbf{r}_1^\uparrow, \dots, \mathbf{r}_{n^\downarrow}^\downarrow) = \sum_k \omega_k \Big(&\det\left[\phi_i^{k\uparrow}(\mathbf{r}_j^\uparrow; \{\mathbf{r}_{/j}^\uparrow\}; \{\mathbf{r}^\downarrow\})\right] \\ &\det\left[\phi_i^{k\downarrow}(\mathbf{r}_j^\downarrow; \{\mathbf{r}_{/j}^\downarrow\}); \{\mathbf{r}^\uparrow\};\right]\Big). \end{aligned}$$

￼

You com



ferminet.png

Motivated for the antisymmetry and the Kato cusp conditions our **Ansatz** take the form of: [

Transformers

There exist several architectures that I can use Recurrent Neural Network, Long Short Term Memory.

Vaswani et al. (2017)

Recurrent Neural Network are: Recurrent Neural Network
And long short term memory are: Long Short Memory

Why on earth I would use Transformer? They are extremely good finding relations between its elements. And the best is that scale well due its Transform Architecture

Attention mechanism appear with Bahdanau et al. (2014) but it didn't work so:

- Attention mechanism
- Self attention mechanism on one head
- Multi-head attention

$$\mathbf{o}_{t,i} = \sum_{j=1}^{t} \text{Softmax}\left(\frac{\mathbf{q}_{t,i}^T \mathbf{k}_{j,i}}{\sqrt{d_h}}\right)\mathbf{v}_{j,i}$$

$$\mathbf{u}_t = W^O[\mathbf{o}_{t,1};\mathbf{o}_{t,2};\ldots;\mathbf{o}_{t,n_h}]$$

Psi Former

Psi Former Ansatz. von Glehn et al. (2023)

$$\Psi_\theta(\mathbf{x}) = \exp(\mathcal{J}_\theta(\mathbf{x})) \sum_{k=1}^{N_{\text{det}}} \det[\mathbf{\Phi}_\theta^k(x)]$$

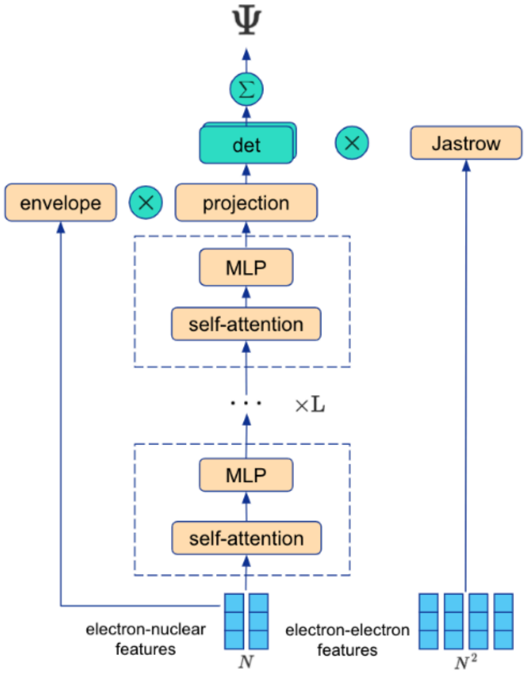
Where $\mathcal{J}_\theta$ is the Jastrow Factor for si Former and Φ are orbitals.

Where $\mathcal{J}_\theta : (\mathbb{R}^3 \times \{\uparrow, \downarrow\})^n \to \mathbb{R}$

- So the question is how you define the outputs of that functions:
- Jastrow Factor

$$\mathcal{J}_\theta(x) = \sum_{i<j;\sigma_i=\sigma_j} -\frac{1}{4}\frac{\alpha_{par}^2}{\alpha_{par} + |\mathbf{r}_i - \mathbf{r}_j|} + \sum_{i,j;\sigma_i\neq\sigma_j} -\frac{1}{2}\frac{\alpha_{anti}^2}{\alpha_{anti} + |\mathbf{r}_i - \mathbf{r}_j|}$$

Architecture



psiformer.png

Loss function

We are going to take the Rayleigh Quotient like Expectation Value like loss function.

Optimizer

Kroenecker factored Approximate Curvature

Flow of the architecture

First compute:

$$v_h = [\mathrm{SelfAttn}(\mathbf{h}_1^l, \ldots, \mathbf{h}_N^\ell; \mathbf{W}_q^{\ell h}, \mathbf{W}_k^{\ell h}, \mathbf{W}_v^{\ell h})]$$

Start with:

$$\mathbf{W}_o^\ell \mathrm{concat}_h [\mathrm{SelfAttn}(\mathbf{h}_1^l, \ldots, \mathbf{h}_N^\ell; \mathbf{W}_q^{\ell h}, \mathbf{W}_k^{\ell h}, \mathbf{W}_v^{\ell h})]$$

With it you can obtain you hidden states, and then how you use it

With them you create the orbital for neural network fermi net

And you have it.

Methodology

To implement the code, the choose of the library is important.

The three options to implement this kind of matter are JAX, Tensor Flow and pytorch, each one with his advantages and disadvantages.

Environment

For this project we are going to be using Pytorch due his user-friendly and support. Python. with UV

Training

Due the high computational power needed we are going to using GPUS and of course CUDA.

Is clear that we are going to use virtual GPUS, for that matter we have two option or well use a GPU via SSH or directly using services like Azure , Colab, or anothers matters.

The election of the GPU is not trivial. use TPUS are not a bad idea.

References

Bahdanau, D., Cho, K., & Bengio, Y. (2014). Neural machine translation by jointly learning to align and translate. *arXiv Preprint arXiv:1409.0473*.

Dosovitskiy, A., Beyer, L., Kolesnikov, A., Weissenborn, D., Zhai, X., Unterthiner, T., Dehghani, M., Minderer, M., Heigold, G., Gelly, S., Uszkoreit, J., & Houlsby, N. (2021). *An image is worth 16x16 words: Transformers for image recognition at scale*. https://arxiv.org/abs/2010.11929

Jumper, J., Evans, R., Pritzel, A., Green, T., Figurnov, M., Ronneberger, O., Tunyasuvunakool, K., Bates, R., Žídek, A., Potapenko, A., & others. (2021). Highly accurate protein structure prediction with AlphaFold. *Nature*, *596*(7873), 583–589.

Luo, D., & Clark, B. K. (2019). Backflow transformations via neural networks for quantum many-body wave functions. *Physical Review Letters*, *122*(22). https://doi.org/10.1103/physrevlett.122.226401

Pfau, D., Spencer, J. S., Matthews, A. G. D. G., & Foulkes, W. M. C. (2020). Ab initio solution of the many-electron Schrödinger equation with deep neural networks. *Physical Review Research*, *2*(3). https://doi.org/10.1103/physrevresearch.2.033429

Qiao, Z., Welborn, M., Anandkumar, A., Manby, F. R., & Miller, T. F. (2020). OrbNet: Deep learning for quantum chemistry using symmetry-adapted atomic-orbital features. *The Journal of Chemical Physics*, *153*(12). https://doi.org/10.1063/5.0021955

Raissi, M., Perdikaris, P., & Karniadakis, G. E. (2019). Physics-informed neural networks: A deep learning framework for solving forward and inverse problems involving nonlinear partial differential equations. *Journal of Computational Physics*, *378*, 686–707. https://doi.org/10.1016/j.jcp.2018.10.045

Shang, H., Guo, C., Wu, Y., Li, Z., & Yang, J. (2025). Solving the many-electron Schrödinger equation with a transformer-based framework. *Nature Communications*, *16*(1), 8464. https://doi.org/10.1038/s41467-025-63219-2

Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., Kaiser, L., & Polosukhin, I. (2017). Attention is all you need. *Advances in Neural Information Processing Systems (NeurIPS)*, *30*.

von Glehn, I., Spencer, J. S., & Pfau, D. (2023). *A self-attention ansatz for ab-initio quantum chemistry*. https://arxiv.org/abs/2211.13672

Excerpt

Transformers are monsters finding relations between what you give them. Is tempting use them for emulate the relations between electrons and protons. How you can first encode the information of the electron's positions and second the attraction and repulsion between the particles?