Fitting Network and Energy Evaluation

Sandip Kumar Sah

Fitting Network and Energy Evaluation

After obtaining the per-atom descriptors $\boldsymbol{\mathcal{D}}_i$ from the descriptor formation stage, the next step is to compute the corresponding atomic energies E_i using a **fitting (energy) network**. Each atomic species (center type) t_c has its own neural network with parameters extracted from the trained DeepMD model.

The total potential energy of the configuration is then

$$E_{\text{total}} = \sum_{i=1}^{N_{\text{loc}}} E_i.$$

Since only a single configuration (frame) is simulated in HALMD at a time, the computation proceeds per atom in that configuration.

Network Inputs and Structure

For each atom i:

- Input descriptor: $\boldsymbol{\mathcal{D}}_i \in \mathbb{R}^{M_1 M_2}$
- Atom type: $t_c(i) \in \{0, \dots, T-1\}$
- Fitting network parameters for type t_c:

$$\{\mathbf{W}^{(l,t_c)}, \mathbf{b}^{(l,t_c)}, \texttt{idt}^{(l,t_c)}\}_{l=0}^{L_t}$$

- Optional per-type bias: b_{t_c}

Each fitting network $f_{\text{fit}}^{(t_c)} : \mathbb{R}^{M_1 M_2} \to \mathbb{R}$ maps the local descriptor to a scalar energy value.

Feedforward Computation with Residual Timestep Connections

Let $\mathbf{x}^{(0)} = \boldsymbol{\mathcal{D}}_i$. The network consists of L hidden layers and one final output layer. For each hidden layer $l = 0, 1, \dots, L-1$:

1. Linear transformation:

$$\mathbf{z}^{(l)} = \mathbf{x}^{(l)}\mathbf{W}^{(l)} + \mathbf{b}^{(l)}$$

2. Nonlinear activation:

$$\phi(\mathbf{z}^{(l)}) = \tanh\big(\mathbf{z}^{(l)}\big)$$

3. Residual–dt update rule: The updated hidden state depends on whether the layer contains a learnable residual timestep parameter `idt`:

$$\mathbf{x}^{(l+1)} = \begin{cases} \mathbf{x}^{(l)} + \texttt{idt}^{(l)} \odot \tanh\big(\mathbf{z}^{(l)}\big), & \text{if } \texttt{idt}^{(l)} \text{ exists,} \\ \tanh\big(\mathbf{z}^{(l)}\big), & \text{otherwise.} \end{cases}$$

Here, $\odot$ denotes element-wise multiplication.

This formulation, known as the **ResNet-dt mechanism**, allows each layer to take a small, learnable "step" in function space, effectively improving training stability and convergence.

Output Layer

The final (linear) layer produces the scalar atomic energy:

$$E_i^\star = \mathbf{x}^{(L)}\mathbf{W}^{(\text{final})} + \mathbf{b}^{(\text{final})},$$

where $\mathbf{W}^{(\text{final})} \in \mathbb{R}^{d_L \times 1}$ and $\mathbf{b}^{(\text{final})} \in \mathbb{R}$.

If the model includes a per-type atomic energy bias, the final energy for atom i becomes

$$E_i = E_i^\star + b_{t_c(i)}.$$

Total Energy Evaluation

Once the individual atomic energies are obtained, the total potential energy of the configuration is simply

$$E_{\text{total}} = \sum_{i=1}^{N_{\text{loc}}} E_i.$$

In HALMD, this sum is performed across all local atoms after evaluating the network for each atom type.

Implementation Notes

The computation follows exactly the logic of the following pseudocode:

```
x = D_i  # descriptor
for layer in layers:
    z = x @ W + b
    if "idt" in layer:
        x = x + layer["idt"] * np.tanh(z)
    else:
        x = np.tanh(z)
E_i = x @ W_final + b_final
if bias_atom_e is not None:
    E_i += bias_atom_e[t_c]
E_total += E_i
```

The weight matrices $\mathbf{W}$ are stored in row-major order, so the dot product `x @ W` corresponds to the operation $\mathbf{xW}$ in the mathematical expressions above.

Shape of Variables

Quantity	Symbol	Shape	Description
Descriptor vector	$\boldsymbol{\mathcal{D}}_i$	$(M_1M_2,)$	Input to fitting network
Hidden activations	$\mathbf{x}^{(l)}$	$(d_l,)$	Layer activations
Weight matrix	$\mathbf{W}^{(l)}$	(d_l, d_{l+1})	Network weights
Bias vector	$\mathbf{b}^{(l)}$	$(d_{l+1},)$	Network biases
Residual step	$\texttt{idt}^{(l)}$	$(d_{l+1},)$	Learnable residual scaling (if any)
Atomic energy	E_i	Scalar	Energy contribution of atom i
Total energy	E_{total}	Scalar	System potential energy

Intuitive Explanation

The descriptor $\boldsymbol{\mathcal{D}}_i$ encodes the structural and chemical environment of atom i. The fitting network performs a nonlinear regression from this descriptor space to the corresponding atomic energy contribution E_i. By summing over all atoms, the total potential energy approximates the potential energy surface of the system:

$$E_{\text{total}} = \sum_i f_{\text{fit}}^{(t_c(i))}(\boldsymbol{\mathcal{D}}_i).$$

The residual timestep connections (`idt`) act like a learnable "integration step," ensuring that information flows smoothly through deeper networks and reducing vanishing gradient effects.