**Response to the Reviewers**

We are grateful to the Reviewers for their valuable suggestions and comments. We have carefully considered all comments and suggestions and have revised the manuscript accordingly. The following details the changes made and brief responses to the questions raised. All the significant changes are marked yellow in the revised manuscript.

**Reviewer(s)' Comments to Author:**

**Reviewer: 1**

The manuscript presented an effort to vectorize the formulism of EAM potential for making use of the modern ML training framework (GPU, Adam optimizer, AutoGrad, ...). The effectiveness of the strategy is overall well-tested, and the findings can be of interest to the MD simulation community. I would be happy to recommend the publication, if the authors could clarify some key issues about the results:   
  
1. The performance of the current ML-EAM/ADP are compared to that of Zjw04 and SNAP. However, it is somewhat unfair to compare with Zjw04, as Zjw04 is trained on a different set of reference data (which can be derived using different calculation settings), thus would naturally have larger MAEs on the current test set.   
  
For example, are there any systematic errors when applying Zjw04 to the test set? For example, most of the energies are prone to be smaller/larger?

It can be better to directly train an EAM potential on the same training data and see if the performance of the current ML-EAM/ADP is similar to the traditional form. 

***Authors’ response:*** *Thanks very much for your comments.*

*In fact, the reference EAM potentials used in this work, developed by Zhou et al, were originally fitted with experiment data. It’s indeed unfair to compare them with potentials optimized with DFT data directly.*

*Hence in our work, relative mean absolute errors (rMAE) were used for the reference EAM potentials, while raw mean absolute errors were used for machine learning optimized potentials. The relative mean absolute errors were calculated by setting the ‘zero point’ to the equilibrium structures of fcc Ni and bcc Mo. This comparison approach was first used by Shyue Ping Ong in their SNAP works1-2.*

2. The authors argued that the MD simulations performed by ML-EAM/ADP is 1000 times faster than SNAP, but it seems no evidence is provided in the manuscript. Are the tests using the same hardware and how are the tests performed? More details should be provided. 

***Authors’ response:*** *Thanks very much for your comments.*

*Thompson et al (core developer of the SNAP) from Sandia National Laboratory already did benchmark tests in 2017 and published the results in a standalone report3. The following figure shows the results:*

Chart, line chart

Description automatically generated

*ADP can be considered an extension of EAM as it only introduces two polarization terms. Lammps official benchmark tests suggest ADP is 3x slower than EAM (https://lammps.sandia.gov/bench.html).*

*ML-EAM/ADP just optimize parameters of EAM/ADP, while their forms are kept.*

*We have made our statement clearer in Section 4.3.*

3. It would be good if the authors provide more details on the GPU version they used and how long it takes to train the potential. Such information may help the beginners to start. 

***Authors’ response:*** *Thanks very much for your comments. We have added more description in our revised manuscript.*

*The training time heavily depends on the settings.*

*Generally speaking, training with energy, forces and stress is very fast. In this case, a few hours are sufficient for a workstation with a single GTX 1080Ti for 100,000 training steps.. Including equation of state constraints will only slightly increase training time. However, adding elastic constraints will significantly increase training time from a few hours to 20 or more.*

*In our experience, a two-steps strategy works well: training with energy, forces, stress and EOS for 50,000 to 100,000 steps first, then add elastic losses to the total loss and restart the training from a checkpoint for 20,000 to 50,000 steps.*

4. On Page 8, the authors argued that "Just like other machine learning tasks, the mini-batch stochastic gradient descent algorithm is used to minimize the loss function. The Adam optimizer [52] is used to minimize Equation 34." I am afraid that mini-batch and SGD, SGD and Adam are different implementations. It would be good if the authors can double-check the technical details.

***Authors’ response:*** *Thanks very much for your comments. We have corrected our statement.*  
  
5. Some typos exist. For example, on page 2 paragraph 1, "calcualte" should be "calculate".

***Authors’ response:*** *Thanks very much for your comments. We have fixed this typo.*

**References**

1. Li, X. G.; Hu, C.; Chen, C.; Deng, Z.; Luo, J.; Ong, S. P. Quantum-accurate spectral neighbor analysis potential models for Ni-Mo binary alloys and fcc metals. *Phys Rev B* **2018,** *98*, 094104.

2. Chen, C.; Deng, Z.; Tran, R.; Tang, H.; Chu, I.-H.; Ong, S. P. Accurate force field for molybdenum by machine learning large materials data. *Phys. Rev. M* **2017,** *1*.

3. Trott, C.; Thompson, A. P. *A Brief Description of the Kokkos implementation of the SNAP potential in ExaMiniMD*; 2017.