A. Empirical Potentials

Empirical potentials are approximations of the potential energy surface for physical systems. In contrast to Density-Functional Theory which computes energies from first principles (Hohenberg & Kohn, 1964), empirical potentials are generic functional forms whose parameters are fitted to correspond to individual systems and which are designed to be efficient to compute. In our paper, these empirical potentials enable the training of learned optimizers, as we can easily batch computation and use automatic differentiation techniques to quickly calculate gradients/forces.

Moreover, the minima of empirical potentials are likely to generalize to those found by more-accurate computations, suggesting that the learned optimizers trained on these approximations will generalize as well. In this paper, we study a few functional forms of empirical potentials:

A.1. Lennard-Jones Clusters

First, the Lennard-Jones Clusters are the archetype of a simple-to-compute potential and are often used to model spherically-symmetric particles or atoms in free-space (a perfect vacuum with no other particles in the entire system). For example, this empirical potential can be used to model nobel gasses or methane (Jones & Chapman, 1924). The energy landscape is defined only by pairwise distances between particles, denoted as d_{ij} for atoms i, j.

$$\sum_i \sum_{j>i} \epsilon \left[\left(\frac{d_0}{d_{ij}} \right)^{12} - 2 \left(\frac{d_0}{d_{ij}} \right)^{6} \right] \tag{5}$$

where ϵ describes the minimum two-particle energy and d_0 describes the distance where this occurs. Following prior work, in our paper we set ϵ and d_0 to 1, as the resulting minima structures can be scaled as necessary for systems where these settings do not hold.

While this model of atoms may seem simple, the corresponding optimization problem is anything but. For the corresponding task with 75 atoms, common gradient-based techniques such as Adam or FIRE cannot obtain the global minimum value and even the best trial out of 150 random initializations has an error greater than 7eV. Please see the main body of the paper for additional discussion of the difficulties of optimizing Lennard-Jones Clusters.

A.2. Gupta Clusters

The Gupta empirical potential adds an additional layer of complexity when modelling the energy of atomic structures. Designed to model lattice relaxations at a metal surface, the Gupta model provides an improved approximation by including a second-moment estimate of the tight-binding Hamiltonian (Gupta, 1981). The Gupta potential has been widely used in studying and predicting the stable structures of noble metals and bimetallic clusters.

The functional form of the Gupta potential is as follows:

$$\sum_i \sum_{j>i} A \exp\left[p \left(1 - \frac{d_{ij}}{d_0} \right) \right] - \xi \sum_i \sqrt{ \sum_{j>i} \exp\left[2q \left(1 - \frac{d_{ij}}{d_0} \right) \right] } \tag{6}$$

The values d_0, A, ξ, p, q are parameters of the Gupta potential that describe specific inter-particle interactions. In contrast to the Lennard-Jones system, these values cannot be factored out or set to defaults. Instead, the parameters are dervied either experimentally or by fitting to Density-Functional Theory data from the bulk-faced cubic systems. Note, when the system consists of only one type of atom, the parameters are the same for all pairs of particles. However, for bimetallic clusters with multiple types of atoms, these parameters can depend on the type of interaction. For example, in Ag-Au clusters, there will be 3 possible values for each of these constants corresponding to Ag-Ag, Ag-Au, and Au-Au interactions.

As the minima structures of the Gupta potential can vary based on the exact parameter values used, all of our experiments are based on the already-discovered configurations by Paz-Borbón et al. (2008). Table 4 provides the parameters used in our single-atom experiment modelling Au (gold) with 55 atoms and bimetallic experiments of Ag-Au, Ag-Pt, Pd-Au, and Pd-Pt clusters.

A.3. Stillinger-Weber (SW)

Stillinger-Weber potentials are designed to provide more accurate estimations of semiconductors and do so by including a three-body angular term (i.e. penalizing for deviations from an optimal angle within the crystal structure). In our paper, we use the Stillinger-Weber potential to model Silicon crystals. This system is distinct from the other two benchmarks, as the crystal model assumes the lattice structure is repeatedly infinitely in space (although a cutoff in interaction distance allows for models to use finite tilings).

For this system, the energy is defined by:

$$\sum_i \sum_{j>i} \theta_2(d_{ij}) + \sum_i \sum_{j\neq i} \sum_{k>j} \theta_3(d_{ij}, d_{ik}, \theta_{ijk}) \quad (7)$$

$$\theta_2(d_{ij}) = A\epsilon \left[B \left(\frac{\sigma}{d_{ij}} \right)^p - \left(\frac{\sigma}{d_{ij}} \right)^q \right] \exp\left[\frac{\sigma}{d_{ij} - a\sigma} \right]$$

$$\theta_3(d_{ij}, d_{ik}, \theta_{ijk}) = \lambda\epsilon \left(\cos\theta_{ijk} - \cos\theta_0 \right)^2 \exp\left[\frac{\gamma\sigma}{d_{ij} - a\sigma} \right]$$

where again the scalar parameters are fit to the system being studied. For modelling Silicon, the parameters used in our experiments are provided in Table 5.

Additionally, we only study the optimization problem of a simple cubic lattice structure, where the lattice vectors are defined by $\vec{i}, \vec{j}, \vec{k}$. Cells of 8 atoms are initialized to have size 5.248 Å, and there is no lattice vector optimization during the course of training. Furthermore, we define the cutoff for atomic interactions to be 3.77 Å.

Table 4. Gupta Potential Coefficients

	P	Q	d_0	A	ξ
Au (Gold) 55					
Au-Au	10.229	4.036	2.884	1.790	0.2061
Bimetallic Ag-Au					
Ag-Ag	10.85	3.18	2.8921	1.1895	0.1031
Ag-Au	10.494	3.607	2.8885	1.4874	0.1488
Au-Au	10.139	4.033	2.885	1.8153	0.2096
Bimetallic Ag-Pt					
Ag-Ag	10.86	3.18	2.8921	1.1895	0.1031
Ag-Pt	10.73	3.57	2.833	1.79	0.175
Pt-Pt	10.612	4.004	2.7747	2.695	0.2975
Bimetallic Pd-Au					
Pd-Pd	10.867	3.742	2.7485	1.718	0.1746
Pd-Au	10.54	3.89	2.816	1.75	0.19
Au-Au	10.299	4.036	2.884	1.79	0.2061
Bimetallic Pd-Pt					
Pd-Pd	10.867	3.742	2.7485	1.718	0.1746
Pd-Pt	10.74	3.87	2.76	2.2	0.23
Pt-Pt	10.612	4.004	2.7747	2.695	0.2975

B. Comparison of Training Strategies

In order to compare the meta-training strategies of ES, ESMC (ours) and GA (ours), we focus on a simplified set-up of the learned optimizers. Specifically, to remove the noise originating from the meta-training on diverse tasks, we focus on models trained only on the 13-atom Lennard-Jones clusters. As the model is trained only on 1 task, we only need to train for 900 meta-updates before both ESMC (ours) and GA (ours) appear to reach the global minimum on almost every single initialization. In contrast, traditional ES appears to be unstable, deviating greatly in the meta-training loss.

Table 5. Stillinger-Weber Potential Coefficients

	A	ϵ	B	P	λ	γ	σ	A
Silicon Crystals								
Si-Si	7.049	2.168	0.602	4	21.0	1.2	2.0951	1.8

As the loss landscape of this system is 'funneled' and a low-energy paths exist between the local minima and the global minimum, we do not expect sporadic behavior of the learned optimizer to arise from the optimization problem itself. Instead, the erratic behavior of appears to be coming from instability in training, which is solved by our ESMC and GA training strategies.

C. Baseline Optimizer Tuning

The baseline optimization techniques used in this paper (Adam, FIRE, Basin Hopping) have their own hyper-parameters which require tuning to obtain proper results. Doing so correctly is essential to ensure that our *learned optimizer* provide a meaningful improvement in minimum discovery that cannot be explained by improved tuning.

C.1. Tuning Adam and FIRE

In our paper, all baseline results arise from a two-stage process. First, we start with a grid search: utilizing 3 variants of the learning rate for each model, with values of $\{0.01, 0.005, 0.001\}$. Early exploration also modified the β parameters of Adam and the rate of increase/decrease of FIRE; however, both optimizers showed generic robustness to hyper-parameters other than learning rate changing.

While these baselines were somewhat competitive, we further tuned these optimizers by *learning* the values of the hyper-parameters for Adam and FIRE, following the *Adam4p* strategy by Metz et al. (2019b). This strategy, similar in spirit to the *learned optimizers* presented in the paper, uses meta-training to update the scalar parameters that define Adam and FIRE. 3 runs were conducted, initializing with each of the learning rates in the grid search. Meta-optimization was conducted for 100 outer-steps, applied with Adam with a learning rate of 10^{-2}. The best hyper-parameters out of the grid search and the final *learned* parameters were then used to provide evaluation results.

C.2. Tuning Basin Hopping

Recall, Basin Hopping first uses standard optimization technique such as Adam or FIRE to optimize a network for a short number of steps. Parameters are perturbed from the minimum found, and optimization is performed once again to find a new minimum. If the new minimum is an improvement over the previous, then the new state is accepted;

otherwise, the model reverts to the previous minimum.

This two-stage optimization technique has a number of hyper-parameters that require tuning. First, to simplify the setup, we fix Adam with a learning rate of 10^{-2} to be the standard optimizer used to descend into minima after the large perturbation steps. Experimental evidence showed that this large learning rate model allowed us to decrease the number of steps to 5000, while almost always converging to a local minima. The final parameter of significance is the size of the step taken, which is drawn from a normal distribution. Similar to the approach to tuning Adam and FIRE, we started with grid search, finding the best values out of the set $\{0.2, 0.4, 0.6, 0.8\}$. Additional tuning was performed by learning the size of the update step. As before, the best of these hyper-parameters was used in reporting results.

D. Training Details

For the sake of reproducible results, we provide additional details about learned optimizer training, including descriptions of how tasks are selected for the meta-batches and about what it means to produce a *random* initialization for particles.

D.1. Task Selection

During meta-training, estimates of the meta-gradients are produced by averaging over ~ 80 instantiations of atomic structure optimization problems, defined by both a random initialization (see Appendix D.2) and a corresponding empirical potential to minimize. Prior work on learned optimizers would refer to this set of problems as a task distribution and sample 80 tasks used to compute the a single meta-update. However, as the learned optimizers trained in this paper are still in the few-task regime, we instead default to sampling $\lfloor 80/m \rfloor$ copies of all m tasks.

D.2. Harmonic Initialization

In the context of atomic structure optimization, purely random initializations (i.e. uniform over a pre-defined box size) are problematic as atoms that are too close to one another will have very large forces early in optimization. This can result in one atom being moved far away from the others. As most molecular dynamics simulations (including ours) use a cutoff distance for atomic interactions, future optimization steps are unlikely to update and recover this atom. As more particles often corresponds to lower energies, the resultant structure will be worse off than initializations where all particles are incorporated into the final structure.

A number of strategies have been proposed to stably initialize sets of atoms or particles. One strategy is to initialize uniformly over a large box, large-enough so that the particles are unlikely to be close but small enough that cutoffs are not a problem. In practice, we found this method difficult to tune when working with a variety of systems.

Instead, we utilize *harmonic initialization*. This strategy starts by randomly initializing coordinates in a small box, of size 3.0 Å for all Lennard-Jones models. Before atomic structural optimization begins, we first optimize an soft-sphere potential, which only penalizes per-particles distances when atoms are within pre-defined cutoff. Optimizing this intermediate function ensures that structural optimization does not have excessively large forces at the beginning of training. In free-space, this occurs by simply spreading particles apart.

A functional form of the soft-sphere potential is provided below:

$$\sum_i \sum_{j>i} \begin{cases} 0 \text{ if } d_{ij} > 0.1 \\ \epsilon \frac{(1-d_{ij})^\alpha}{\alpha} \text{ otherwise} \end{cases} \tag{8}$$

where in our formulation $\epsilon = 1$ and $\alpha = 2$. By default, we use 1000 optimization steps performed by gradient descent with a learning rate of 10^{-3}. Note, the learned optimziers are not sensitive to changes in the harmonic step count or learning rate; both defaults were chosen to allow excess time for the soft-sphere potential to be minimized to ≈ 0.

The same strategy is used for Gupta potentials, with the only difference being that the box is increased to edge length 4.0 Å to accomodate for the increased size of the atoms. For crystal optimization with the Stillinger-Weber potentials, our initialization respects the periodic boundary conditions, so particles are optimized within the pre-defined lattice.

D.3. Learned Optimizer Initialization

All weights of the neural network used to parameterize the learned optimizer are initialized via a LeCun Normal initialization (LeCun et al., 2012), following the default in the FLAX library (Heek et al., 2020), except for the output layer which is initialized to have output variance of 0. This default ensures that the learned optimizer at the beginning of training does not yield divergent trajectories.

For meta-training, the parameters that have the most significant impact on performance are the α, β, γ used in the output parameterization. Best models used

$$\alpha = 0.1 \qquad \beta = 1 \qquad \gamma = -3$$

We hypothesize that this setup is stable when combined with the 0 output initialization, as steps start out very small and increase in size over the course of meta-training. To large of a step early in training may yield the undesirable scenario of particles becoming too close to one another.

Table 6. Transfer performance between potential types show that learned optimizers trained on Leannard-Jones can generalize to Stillinger-Weber, outperforming Adam and FIRE. However, the reverse is not true as the models trained on Silicon find very poor local minima, likely due to the removal of the periodic boundary condition.

	Mean of 150 Initializations		
Training Set	Lennard-Jones 13 atoms	Lennard-Jones 75 atoms	SW Silicon 64
Baselines			
Adam	-40.6	-380.5	-256.8
FIRE	-40.5	-380.2	-257.0
Learned Optimizers			
Lennard-Jones {13, 19, 31, 38 55, 75}	-44.3	-390.3	-261.7
Silicon 64	-35.6	-258.3	-261.8
Global Minimia	-44.3	-397.5	-277.2

D.4. Compute Costs

Training costs vary significantly based on the system and distributed setup. For example, costs scale quadratically in the number of atoms for the Lennard-Jones and Gupta cluster and cubicly for the Stillinger-Weber models due to the three-body terms. Rough estimates of the training time are 30 GPU hours for the Lennard-Jones models (single atom type) with the main bottleneck coming from the optimization of the 75 atom system. The Gupta models took about 10 GPU hours due to the smaller size.

E. Implementation

As mentioned in the core body of the text, the empirical potentials make use of Jax_MD (Schoenholz & Cubuk, 2019) and the optimization makes use of pure JAX (Bradbury et al., 2018). Code is available: `https://learn2hop.page.link/github`.

F. Additional Results

In this section, we present additional results that were unable to fit in the main body of the paper. These results including minimal training task distributions and results beyond atomic structure optimization further support the use of learned optimizers in these global minimization problems.

F.1. Training with a Single Task

The results presented in Section 4 show significant benefits on global minima discovery when training only on a subset of 6 optimization tasks (defined by a different number of Lennard-Jones atoms). This diverse training set provides examples of both simple 'funneled' landscapes and glassy landscapes with large energy barries between minima. However, it may be possible to perform well with significantly fewer training tasks.

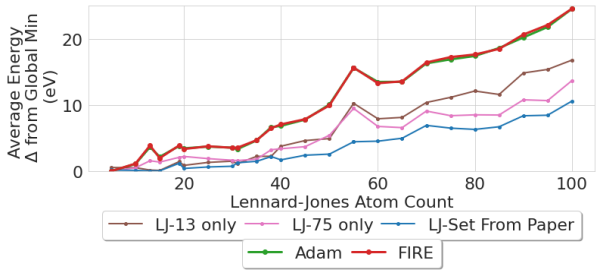

Figure 8. Results for the Lennard-Jones single atom models, replicated with various training sets (13-atoms only, 75-atoms only, or a set of 6 different counts). Both models trained only on 1 atom count show improvements over Adam or FIRE but lack the generalization performance of training from a diverse set.

In Figure 8, we show a comparison between training the learned optimizer on only Lennard-Jones with 13 atoms (LJ-13), Lennard-Jones with 75 atoms (LJ-75), and the diverse Lennard-Jones set from the main body of the paper {13, 19, 31, 38, 55, 75}. Interestingly, the models trained only on 13 or 75 atoms often perform better than Adam and FIRE and generalize significantly beyond the respective training distributions. The 13 atom model is perhaps the most impressive as it is the cheapest to train (due to the quadratic slowdown with number of atoms); but the model taking into account diverse examples show greater generalization to large atom counts.

F.2. Transfer Between Potentials

Early results in Table 6 also show that learned optimizers can transfer between empirical potentials; for example, the models trained on Lennard Jones clusters can transfer to the Stillinger-Weber potential despite differences in the periodic boundary and the addition of the angular term (but not vice

versa). We believe that this form of generalization is most interesting and hope that future work explores this direction further; learned optimizers that train on empirical potential and can be applied to DFT simulation appears a promising avenue for significantly speeding up material design.