A Equivalence of Distributions in the PST

If a PDD is arbitrarily expanded or collapsed, the PST should produce the same results as these PDDs are considered equivalent. The same can be said of any input distribution. This is proven here:

Lemma A.1. *Let A and B be weighted multisets each containing elements from the set $S = \{\boldsymbol{x}_1, \dots \boldsymbol{x}_n\}$. Each element $\boldsymbol{x}_i \in \mathbb{R}^{1\times n}$ occurs with multiplicity $m_i^{(a)} \in \mathbb{N}^+$ and $m_i^{(b)} \in \mathbb{N}^+$ in A and B respectively. Each element also carries weight $w_i^{(a)} \in \mathbb{R}^+$ and $w_i^{(b)} \in \mathbb{R}^+$ for A and B respectively. The application of the Periodic Set Transformer will yield equivalent output if*

$$w_i^{(a)} m_i^{(a)} = w_i^{(b)} m_i^{(b)}, \qquad \forall i \in \{1, \dots, n\} \tag{S1}$$

Proof. To prove that the output of the PST is equivalent for A and B, it is sufficient to prove that the output of σ (defined in Equation (2)) and the pooling layer (defined in Equation (4)) are the same for A and B. Let $\boldsymbol{q}_i = \boldsymbol{x}_i \boldsymbol{W}_Q$, $\boldsymbol{k}_i = \boldsymbol{x}_i \boldsymbol{W}_K$, and $\boldsymbol{v}_i = \boldsymbol{x}_i \boldsymbol{W}_V$ be the query, key, and value vectors produced by the weight matrices $\boldsymbol{W}_Q, \boldsymbol{W}_K, \boldsymbol{W}_V \in \mathbb{R}^{n\times d}$. First, note the pre-softmax attention weight from $\boldsymbol{x}_i$ to $\boldsymbol{x}_j$ can expressed as:

$$a_{ij} = \frac{\boldsymbol{q}_i \boldsymbol{k}_j^T}{\sqrt{d}} \tag{S2}$$

and is independent of both weight and multiplicity. Further, notice that the summation of an expression over A or B can be rewritten using its multiplicities. For A this is done like so:

$$\sum_{t=1}^{|A|} (\cdot) = \sum_{s=1}^{n} m_s^{(a)} (\cdot)$$

Using this equivalence, the function σ, used to calculate the attention weight from $\boldsymbol{x}_i$ to $\boldsymbol{x}_j$ for A and $\mathbb{B}$, is written as:

$$\alpha_{ij}^{(a)} = \frac{w_i^{(a)} exp(a_{ij})}{\sum_{k=1}^{n} w_k^{(a)} m_k^{(a)} exp(a_{ik})} \tag{S3}$$

and

$$\alpha_{ij}^{(b)} = \frac{w_i^{(b)} exp(a_{ij})}{\sum_{k=1}^{n} w_k^{(b)} m_k^{(b)} exp(a_{ik})} \tag{S4}$$

Using the condition provided by Equation (S1), Equation (S4) can be rewritten as:

$$\alpha_{ij}^{(b)} = \frac{w_i^{(b)} exp(a_{ij})}{\sum_{k=1}^{n} w_k^{(a)} m_k^{(a)} exp(a_{ik})} \tag{S5}$$

$$\sum_{k=1}^{n} w_k^{(a)} m_k^{(a)} exp(a_{ik}) = \frac{w_i^{(b)} exp(a_{ij})}{\alpha_{ij}^{(b)}} \tag{S6}$$

Substituting back into Equation (S3) gives us:

$$\alpha_{ij}^{(a)} = \alpha_{ij}^{(b)} \frac{w_i^{(a)} exp(a_{ij})}{w_i^{(b)} exp(a_{ij})} \tag{S7}$$

$$\alpha_{ij}^{(a)} = \alpha_{ij}^{(b)} \frac{w_i^{(a)}}{w_i^{(b)}} \tag{S8}$$

The attention weights produced from A can now be expressed in terms of the attention weights produced from B. The j^{th} entry

in the attention vector for $\boldsymbol{x}_i$ in A is:

$$y_{ij}^{(a)} = \sum_{j=1}^{n} \alpha_{ij}^{(a)} m_j^{(a)} v_{ji} \tag{S9}$$

$$= \sum_{j=1}^{n} \alpha_{ij}^{(b)} \frac{w_i^{(a)}}{w_i^{(b)}} m_j^{(a)} v_{ji} \tag{S10}$$

$$= \sum_{j=1}^{n} \alpha_{ij}^{(b)} \frac{w_i^{(a)}}{w_i^{(b)}} \left(\frac{w_i^{(b)} m_i^{(b)}}{w_i^{(a)}} \right) v_{ji} \tag{S11}$$

$$= \sum_{j=1}^{n} \alpha_{ij}^{(b)} m_i^{(b)} v_{ji} \tag{S12}$$

$$= y_{ij}^{(b)} \tag{S13}$$

Thus, the resulting embeddings from the attention mechanism are equivalent. While the embeddings themselves are equivalent, the cardinality for each embedding still differs according to each multisets' multiplicity. The pooling described by Equation (4) fixes this. Let $\boldsymbol{z}_i$ be the final embedding for $\boldsymbol{x}_i$. The output vector $\boldsymbol{z}$ from the pooling for A is the sum:

$$\boldsymbol{z}^{(a)} = \sum_{i=1}^{n} w_i^{(a)} m_i^{(a)} \boldsymbol{z}_i \tag{S14}$$

Substituting Equation (S1) again we get,

$$\boldsymbol{z}^{(a)} = \sum_{i=1}^{n} w_i^{(b)} m_i^{(b)} \boldsymbol{z}_i \tag{S15}$$

$$= \boldsymbol{z}^{(b)} \tag{S16}$$

Thus, the output of the PST for both A and B are equivalent. □

B Applications of the Pointwise Distance Distribution

In definition 1.2 we provided a formal definition for the PDD. Here, we will begin by providing an example of its construction.

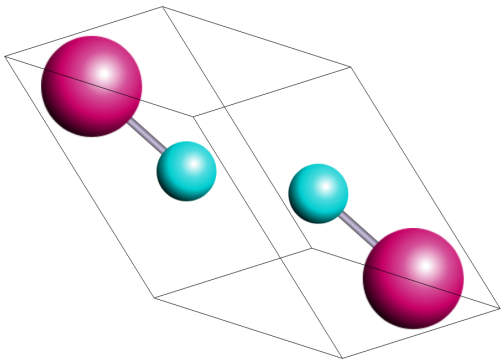


Figure S1. The unit cell of Lutetium-Silicon; Silicon is colored in teal and Lutetium in magenta.

Consider Lutetium-Silicon which is shown in Figure S1. The unit cell pictured contains a total of four atoms, 2 of which are Lutetium and 2 of which are Silicon. If we use $k = 2$ nearest neighbors, the PDD matrix of this periodic set S before rows are grouped is

$$PDD(S;k) = \begin{pmatrix} 0.25 & 2.481 & 2.481 \\ 0.25 & 2.481 & 2.481 \\ 0.25 & 2.881 & 2.881 \\ 0.25 & 2.881 & 2.881 \end{pmatrix}$$

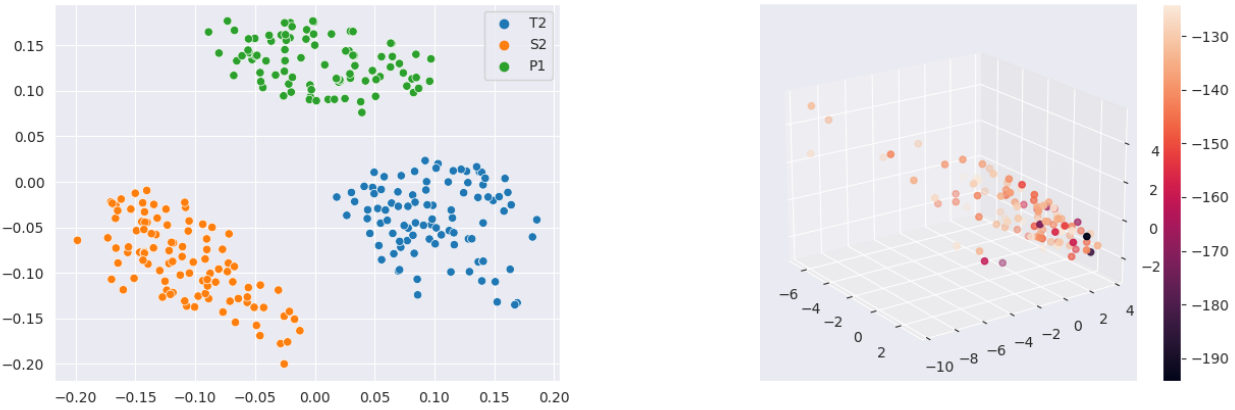
where the first column contains the weights for each row (atom). The second is the distance to the nearest neighbor and the third, the distance to the second nearest neighbor. The first two rows are identical, as are the final two. Because of this, each of the two groups of rows can be grouped into a single row as so,

$$PDD(S;k) = \begin{pmatrix} 0.5 & 2.481 & 2.481 \\ 0.5 & 2.881 & 2.881 \end{pmatrix}$$

The rows are already lexicographically ordered so this is the finalized PDD. If a different unit cell is selected, this collapsing of matrix rows will continue to yield the same result as the proportion of each atom will not change.

The PDDs of two periodic crystals can be compared using the Earth Mover's Distance[6]. The weights of each PDD can be considered the distribution we are comparing in the minimum flow problem. We can visualize a set of crystals by projecting these distances onto a two or three-dimensional plane.

Subfigure S2a shows the projection of the pairwise distances of the PDDs of one hundred samples from each of the three crystals in this experiment using Multi-Dimensional Scaling (MDS). This technique attempts to map a set of pairwise distances to specific n-dimensional space by minimizing the difference between the actual pairwise distances and the distances between the points in the projected space. Even with this error, we are able to establish three distinct clusters in two-dimensional space according to their respective crystal type.



(a) MDS of sampled data by crystal type

(b) MDS of T2 crystals

Figure S2. Multi-dimensional scaling projection on to $\mathbb{R}^2$ and $\mathbb{R}^3$ for the pairwise distances of crystals between each other. Subfigure (a) projects three types of crystals using distances created by using the Earth Mover's Distance between PDDs for $k = 10$. Subfigure (b) is created using the MDS projection of pairwise distances between PDDs at $k = 100$ for one hundred random samples from the T2 crystals colored by lattice energy measured in kJ/mol.

Subfigure S2b shows the projection of the pairwise distances between the crystals' PDD onto three-dimensional space for the T2 dataset. The colors of the points in each plot signify the lattice energy for the crystal. The distinction here is not as pronounced; this is to be expected as the sampled crystals are much more similar in their structure and thus the distances between the PDDs are smaller. Nonetheless, there is a discernible trend, and crystals that congregate near each other do share similar lattice energies.

C Prediction of Lattice Energy

Here, the dataset considered contains simulated molecular crystals created by[54] during the crystal structure prediction using quasi-random sampling[55]. This data is subsetted by the underlying molecule, e.g. T2. During structure prediction, crystals are generated while traversing the potential energy surface and their lattice energy is calculated using *DMACRYS*[56] to determine their stability.

Prediction of lattice energy is done in three different scenarios. In the first, the model will be applied to a single set of molecular crystals with the same composition using $80/10/10$ training, validation, and testing splits. Next, the model is applied to multiple sets of crystals each with different underlying molecules using the same splits. The final experiment consists of the application of the model to a set of crystals with an underlying molecule it has not seen in training. The seen data is split $90/10$

for training and validation. To make predictions the PST described in section 1.1 is used with only the PDD as input and no knowledge of the composition.

Invariants are usually used to discern crystals by measuring differences between their structure. Here, the goal is to demonstrate the effectiveness of using an invariant as a representation for a machine learning algorithm. Even when compositional information is not present, the PDD can distinguish crystals with the EMD from the changes in the pairwise distances that occur when the species of atoms are changed. Whether the same distinction can be made in the context of a learning algorithm has previously not been shown.

We make a comparison to another invariant that has been used to predict lattice energy. Average-minimum-distance[36] was used as input for a Gaussian regression model[37]. In Table S1 we list the performance on the test set of this AMD model to allow comparison between invariants.

Our model reduces the mean-absolute-error (MAE) rate by 21% compared to the Gaussian Process Regression technique which utilizes AMD[37]. The mean-absolute percentage error (MAPE) is also reduced by 1.63%. While we use $k = 60$ nearest neighbors for constructing the PDD, the AMD model uses $k = 500$ to achieve its best results. Despite the use of this additional information, the model using the PDD still performs more accurately.

Table S1. Results of lattice energy prediction using the PDD with the PST and AMD with Gaussian regression on three different tasks using $80/10/10$ training, validation, and testing splits. **(a)** uses only experimentally generated structures of the T2 molecular crystal based on triptycene. **(b)** adds two additional sets of molecular crystals, P1 (based on pentiptycene) and S2 (based on spiro-biphenyl). **(c)** uses P1, P1M (methylated analogue of P1) and P2 (benzimidazolone series pentiptycene) in the training and validation set using a $90/10$ split. The test set consists only of P2M, the methylated analogue of P2.

	Train	Train Size	Test	Test Size	Invariant	MAE (kj/mol) ↓	MAPE ↓
(a)	T2	4,630	T2	578	AMD	4.79	4.31%
					PDD	3.76	2.68%
(b)	T2, P1, S2	14,547	T2, P1, S2	1,819	AMD	4.68	2.83%
					PDD	4.11	2.52%
(c)	P1,P1M,P2	22,995	P2M	7,352	AMD	12.99	6.89%
					PDD	7.24	3.89%

In the second row of Table S1 we list the results of the second experiment. The previous task is extended to a dataset of crystals that contains different underlying molecules. These molecules have different compositions. This compositional information is not contained within the PDD (and thus, not in our input). The model will have to discern crystals solely by their structure. While the overall MAE has increased slightly, the percentage error has decreased. The domain on which the lattice energies lie is different for each type of crystal. Using the PDD alone is enough for the algorithm to distinguish the crystal types and predict lattice energy accordingly.

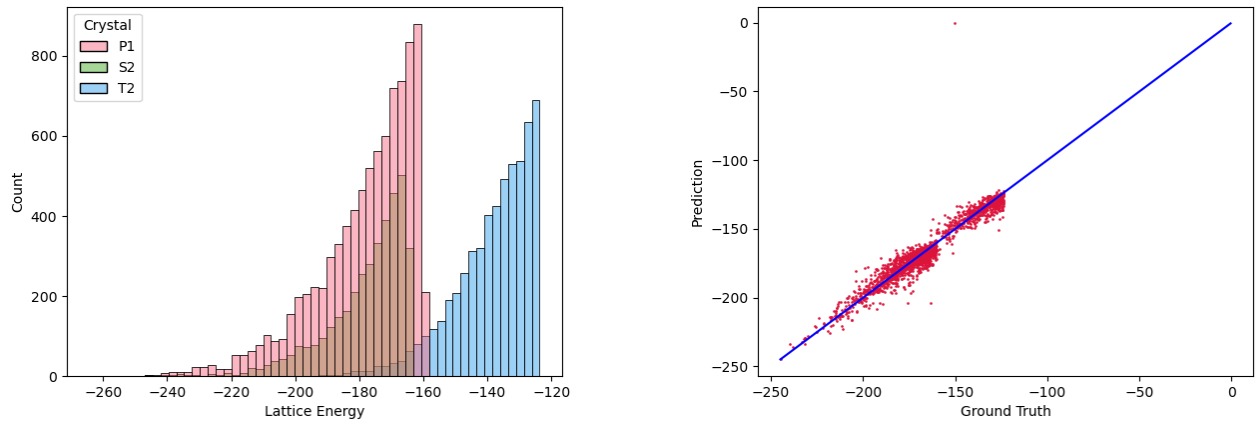
The final experiment uses the data from the P1, P1M, and P2 crystals in the training and validation data. The test set consists of the P2M crystal, which is not part of the either training or validation set. This experiment is the closest to real-world conditions in which new crystals often arise and finding the stable forms is crucial, but information on their lattice energy is unavailable.

When lattice energy is calculated using ab initio calculations, the range of the energies varies from crystal to crystal. When introducing a new type of crystal for our algorithm to make predictions on, this can become a problem as extrapolation to unvisited parts of the lattice energy range can result in high error rates. Fortunately, lattice energies between different types of crystals, are not usually meant to be compared. Instead, they are generated for potential polymorphs of a crystal in an effort to find those with the highest stability for synthesis. We can make use of this fact when applying our model to novel crystals. It is not necessary to predict the correct range of lattice energies; instead, the model needs to be able to predict the lattice energies of the various structures such that their ordering according to their lattice energy is correct. This task could feasibly be turned into a *learning-to-rank* problem[57], but as a regression task, it allows for a more general approach since the predicted lattice energies can be ordered after the fact.

Each dataset has its lattice energies shifted by the mean lattice energy towards zero. By doing this they each maintain their distribution but now overlap around the origin. The model is trained and validated based on this shifted data. The MAE of the predictions on the test set after they have been shifted back is 7.24 kJ/mol and the MAPE is 3.89%.

While the MAE and MAPE are higher than in previous experiments, the improvement over AMD is more significant. The majority of errors come from underestimating the lattice energy. The datasets tend to grow sparser in these areas where lattice energy is lower as this is where the most stable structures tend to lie. Having a false positive (predicting a higher energy structure to be lower) increases the number of potentially stable structures. False negatives are more impactful as they may result in a structure not being considered entirely due to its seemingly low stability.

The histogram in Figure S3a shows the lattice energies (in kJ/mol) of the three crystals within the dataset. Figure S3b shows the comparison of the predictions of the model against the ground truth values.

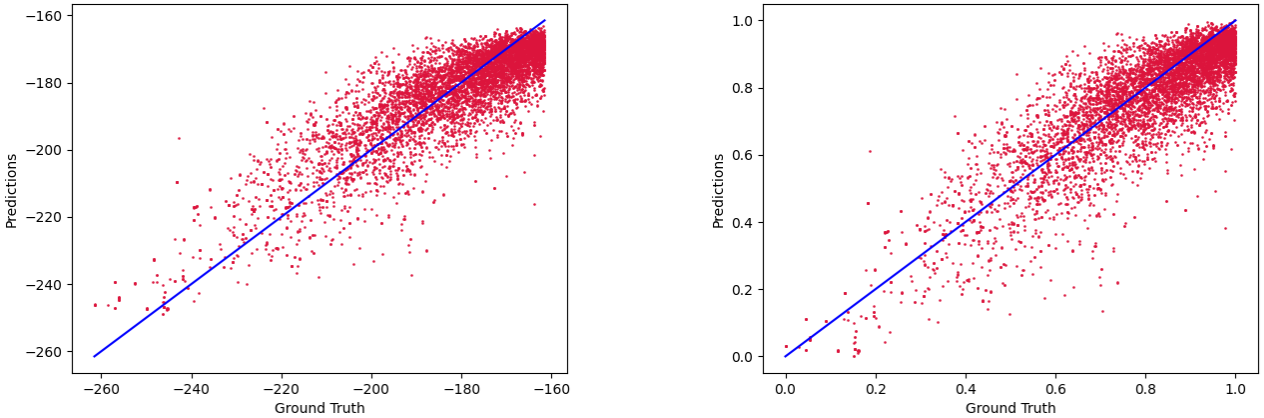


(a) Distribution of ground truth lattice energies.

(b) Ground truth lattice energy vs. predictions

Figure S3. (a) The distribution of the lattice energies of the T2, S2 and P1 crystals. (b) The predictions of T2, S2, and P1 compared to the ground truth lattice energies in kJ/mol.

Predictions that have lower errors will have their point placed closer to the line colored in blue. The bulk of the points share their error both below and above the true lattice energy as we would expect in a model without bias. There are a few outliers, in particular, a single crystal from the T2 dataset has a predicted lattice energy of just -0.41. Prevention of such errors can be mitigated by using a different loss function than MAE. In particular, mean-squared error (MSE) and Huber loss can hedge against outlying errors by increasing their contribution to the loss function. We choose to not present the results using these loss functions as MAE still provides better results for MAE and MAPE.



(a) Ground Truth vs. Predictions re-scaled according to mean.

(b) Ground Truth vs. Predictions normalized between zero and one.

Figure S4. (a) Comparison of predictions vs. ground truth of the test set after the predictions are scaled back by the mean of the lattice energies. (b) The comparison of predictions vs. ground truth after both have been normalized between zero and one.

In Figure S4a we plot the original predictions on P2M after they have been re-scaled back away from the mean lattice energy. In Figure S4b, we re-scale the prediction and the ground truth values to between zero and one. By doing this, we can see the ordering of the predictions compared to the true lattice energies. The scatter plot in Figure S4b allows us to compare lattice energies relative to other predictions.

If the error rate produced by the final experiment of section C is inadequate, we can supplement the training with predictions

from classical methods. For a novel crystal, ab initio calculations can be used to generate lattice energies for a small subset of structures. These samples can be integrated into the training set to improve the overall results. In order to make this practical, we can only produce lattice energies for a limited portion of structures; we limit this to 10% (or under) of the total structures.

Table S2. Effect of including portions of the P2M data into the training set on prediction accuracy.

Samples	Percent of P2M Data	Test MAE (kj/mol) ↓	Test MAPE ↓
0	0%	7.24	3.89%
367	5%	5.82	3.17%
735	10%	5.50	3.02%

Table S2 displays the result of adding this supplemental data to the training set. As expected, the addition of data decreases MAE and MAPE. Even with as few as 367 samples (or 5% of the total dataset), the reduction in error is significant. This process experiences diminishing returns as the amount of the original dataset is used in the training set. Though the MAE and MAPE continue to decrease, it is questionable whether or not spending time generating the instances using classical methods is worth the additional performance gains.

D Additional Experiments

D.1 Effect of k-nearest neighbors

PDD encoding can be said to be parameterized by two values, the collapse tolerance and the number of k-nearest neighbors. The integer k determines the dimensionality of initial PDD encoding embedding. As k increases, it retains all information from the previous (smaller) values of k. The initial nearest neighbor distances are the most important and the embedding has diminishing returns after this. For any value of $k > 1$, the PDD is invariant. Thus, if the PDD is different, the crystals are guaranteed to be structurally different. In order for the PDD to be distinct such that if any two crystals are different, their PDDs are different, we need the property of generic completeness. This property is given provided the lattice L and sufficiently large k. An upper bound on this k is when all distances in the last column of the PDD are larger than twice the covering radius of the lattice L of the periodic set. We would expect the performance of the model to increase up until this point.

The upper bound on k can be exceedingly large so we implement a heuristic to find a lower bound that is more computationally efficient. As k increases, the number of rows collapsed in the PDD will either stay the same or decrease. The lower bound on k can be considered an integer large enough that the groups established at the upper bound of k are the same as this lower bound. Each crystal could have a different k for which this requirement is met. Our encoding method prohibits the use of a dynamic k value, therefore we need a consistent value for k that can be applied to all crystals in the dataset. The results in Table 1 use $k = 15$. At this value of k the previously mentioned lower bound is satisfied for 99.1% within the formation energy dataset. Increases in k past this point cause marginal improvements to this coverage that were deemed insufficient when the increased computational cost is considered.

Property (units)	MAE by k-Nearest Neighbors PDD Encoding ↓			
	5	10	15	20
Band Gap (eV)	0.261	0.232	**0.212**	<u>0.214</u>
Formation $(eV)/atom$	0.039	0.034	<u>0.032</u>	**0.032**
Shear Modulus $log_{10}(GPa)$	0.087	0.077	<u>0.075</u>	**0.075**
Bulk Modulus $log_{10}(GPa)$	0.064	0.059	<u>0.055</u>	**0.056**
Refractive Index	0.318	0.291	<u>0.292</u>	**0.283**
Phonon Peak $1/cm$	**26.38**	27.89	<u>27.74</u>	29.21
Exfoliation $meV/atom$	38.77	33.63	**31.55**	<u>31.70</u>
Perovskites $(eV)/atom$	0.031	0.031	<u>0.030</u>	**0.030**

Table S3. Prediction MAE on the Materials Project crystals for various k-nearest neighbors PDD Encoding at a collapse tolerance of exactly zero. Errors in bold indicate the value of k with the best performance and underlined errors indicate the second-best performance (lower is better ↓).

The error rates listed in Table S3 vary the value of k and report the resulting mean MAE across the five folds. As expected, the lower values of k generally result in higher MAE. Increasing k eventually causes the error rates to stop decreasing. This is also in line with what would be expected as the PDD has enough information to distinguish itself and additional distances

are unnecessary. This is not the case only with phonon peak, however the differences in MAE are relatively small when the deviation of the errors across the folds is considered.

D.2 Effect of collapse tolerance

The collapse tolerance dictates which rows of the PDD will be collapsed. As this parameter increases, the size of the grouped rows will increase. Once rows are grouped, their distances are averaged in the row which represents the group. The change in this averaged row is proportional to the size of the collapse tolerance. In $PDD(S;k)$, as the collapse tolerance approaches infinity, the PDD will decrease in the number of rows until it consists of just a single row with a weight equal to one. In $PDD(S;k)$, the same increase in collapse tolerance will result in a number of rows within the PDD equal to the number of unique elements within the crystal. In both cases, a collapse tolerance that is large enough will result in information loss, eventually increasing errors in predictions.

Property (units)	MAE by Collapse Tolerance in PDD Encoding ↓			
	1.0	10^{-2}	10^{-4}	0.0
Band Gap (eV)	0.241	0.222	<u>0.210</u>	**0.212**
Formation $(eV)/atom$	0.037	0.033	<u>0.032</u>	**0.032**
Shear Modulus $log_{10}(GPa)$	**0.074**	0.074	<u>0.074</u>	0.075
Bulk Modulus $log_{10}(GPa)$	0.056	0.056	**0.056**	<u>0.055</u>
Refractive Index	**0.283**	0.292	<u>0.290</u>	0.292
Phonon Peak $1/cm$	<u>28.41</u>	28.50	29.40	**27.74**
Exfoliation $meV/atom$	32.19	**31.13**	<u>31.15</u>	31.55
Perovskites $(eV)/atom$	0.030	0.030	0.030	0.030

Table S4. Prediction MAE on the Materials Project crystals using PDD Encoding at various collapse tolerances with $k = 15$. Errors in bold indicate the collapse tolerance with the best performance and underlined errors indicate the second-best performance (lower is better ↓).

The collapse tolerance is varied and then applied to the Materials Project crystals. The results of this experiment are listed in Table S4.

By increasing the collapse tolerance and reducing the number of rows within the PDD, we can increase the speed of computations. Thus, it is important to choose a tolerance that is maximal, while not sacrificing accuracy. The impact of the collapse tolerance on the size of representation is listed in Table S5. These values are calculated by dividing the number of rows in the PDD by the number of atoms in the unit cell. The number of atoms in the unit cell is used to determine the number of vertices in the crystal graph[12]. In this way, the size of our representation can be compared to that of popular graph-based models. Data for crystals typically comes in the form of Crystallographic Information Files (CIF). These files also indicate the amount of potential measurement error for the atomic positions. This is used as a guide and a collapse tolerance of 10^{-4} is used in the experiments for the results in Table 1. Sometimes, however, a higher collapse tolerance can act as a regularization technique that is useful on smaller datasets. This effect is seen in the results for refractive index and exfoliation energy, but there is still a balance to be struck. In larger datasets, the performance regression is more noticeable. A collapse tolerance of one is far higher than what would be necessary to cover measurement error in atomic coordinates and would not be advised, even with the potential efficiency gains. Overall, the collapse tolerance does not have a very large impact due to the prevention of rows corresponding to different atoms from being collapsed in the PDD.

E Implementation Details

The Periodic Set Transformer is implemented using *PyTorch*[58]. There is also a version implemented using *Tensorflow*[59], however, we have found this version to significantly underperform when compared to the PyTorch version. We believe this to be due to how the output of individual attention head output is handled in their respective implementations of Multi-head Attention.

Data pre-processing is fairly minimal. Each crystal comes in the form of a `Pymatgen` structure. The structure is converted into a `PeriodicSet` object. This functionality is provided by the *AMD* package[36]. The PDD of each of the `PeriodicSet` objects is then calculated with the desired collapse tolerance and k value. Each column is then normalized to between zero and one. This is not necessary for achieving the desired accuracy but it does significantly improve the speed of training by requiring fewer epochs.

With respect to the results in Table 1, training is done on each property with the same hyper-parameters. The hyper-parameters that govern PDD encoding remain the same for all properties: a tolerance of 10^{-4} and $k = 15$. Training options

Property	Mean $\lvert M \rvert$	Size of Input				Percentage of $\lvert M \rvert$			
		0.0	10^{-4}	10^{-2}	1.0	0.0	10^{-4}	10^{-2}	1.0
Phonon Peak	7.5	7.3	3.6	3.5	3.4	96.8%	55.7%	54.5%	53.8%
Ref. Index	16.9	16.5	6.7	6.2	5.8	97.7%	49.1%	46.3%	44.1%
Bulk Modulus	8.6	8.3	4.1	3.9	3.7	96.7%	62.1%	60.4%	59.4%
Shear Modulus	8.6	8.3	4.1	3.9	3.7	96.7%	62.1%	60.4%	59.4%
Band Gap	30.0	29.2	13.1	12.0	10.1	97.0%	52.0%	48.5%	44.2%
Formation	29.1	28.4	12.7	11.6	9.9	96.9%	53.1%	49.7%	45.7%
Exfoliation	7.2	7.1	3.6	3.3	3.2	98.8%	55.9%	51.9%	51.5%
Perovskites	5.0	4.9	4.6	4.6	4.6	99.0%	94.8%	94.8%	94.8%

Table S5. Size of the input representation for each dataset in the Materials Project at various collapse tolerances at $k = 15$ compared to the number of atoms in the unit cell $|M|$. The size of the input refers to the cardinality of the input set determined by the number of rows in the PDD. The percentage of $|M|$ is the input set's cardinality divided by the number of atoms in the unit cell, expressed as a percentage.

including weight decay, epochs, and learning schedule are kept the same as well, except for batch size. Batch size is either 32 or 64 depending on the number of samples: 32 if the number of crystals in the dataset is less than 5000 and 64 if greater.