Accelerating Material Design with Tensor Completion

Shaan Pakala[1*] **and Evangelos E. Papalexakis**[1]

[1]University of California, Riverside, Department of Computer Science & Engineering, Riverside, CA, USA
[*]shaan.pakala@email.ucr.edu

ABSTRACT

When designing new materials, it is often necessary to design a material with specific desired properties. Unfortunately, it is not always possible to efficiently calculate these material property values. Computational approaches have made great strides in performance, including the use of Density Functional Theory (DFT) and machine learning (ML). ML in particular has shown great promise through using Graph Neural Networks (GNNs). One main limitation of these approaches is the reliance upon the structure of the materials, which may not always be available. For this reason, we focus on composition-based material property prediction, where we use solely the chemical formula to infer material property values. This way we provide a tool for the efficient scanning of many materials for ideal property values, without requiring any structure calculations. Furthermore, we model composition-based property prediction data as tensors, allowing us to use tensor completion algorithms for inference. In our experiments, using tensor completion for composition-based material property prediction shows better performance than alternative baseline ML methods. This is especially true when we are working with limited training data, which tensor completion methods are equipped to handle. By modeling these tasks as tensors, we are also able to predict multiple material properties at once, by creating an additional tensor mode corresponding to the different material properties. Utilizing tensor decomposition, we are able to more efficiently predict material property values using just the chemical formula, without needing as much training data. By efficiently scanning through material property values using just chemical formulas, even with limited training data, tensor completion has the potential to significantly accelerate materials design and become a valuable tool for material science researchers.

1 Introduction

Efficiently designing new materials with optimal properties can be a significant challenge due to the explosive number of combinations as new design variables are added. When searching for new materials with particular material property values, it can be very expensive to exhaustively generate these combinations of materials. For this reason, there is growing interest in being able to predict these material property values, without having to first produce the material. Computational methods have shown great promise in for inferring material property values, allowing material scientists to quickly find materials with optimal properties. This includes the use of methods such as DFT[1–4], ML[5–12], or a combination of both[13]. In this work, we focus on using tensor completion methods to predict selected material property values that are of interest to the material science community. This includes predicting material property values such as **band gap** for semiconductor materials[8,12,14,15], **magnetization** for soft magnetic materials[9,16], **formation energy**[17–20], and **energy above hull** for thermodynamically stable materials[21–23].

There have already been great advances in structure-based material property prediction with DFT and GNNs. However, it is not uncommon for the structure of a material to be unknown and time-consuming to calculate[24,25]. For this reason, we focus on composition-based material property prediction[6,24–27], where we use only the chemical formula as input. This allows material scientists to quickly screen through materials, without needing to initially compute the structure. We set up composition-based material property prediction tasks as tensors to show that tensor completion[28] shows very promising performance and utility for this purpose. This is especially true when we have very limited training data, where tensor completion methods excel. We compare different tensor completion methods with several baseline ML methods, including GNNs[29] and XGBoost[30]. With composition-based material property prediction using limited training data, tensor completion has the ability to efficiently scan through many materials, and potentially significantly accelerate the material design process.

Our overall contributions can be outlined as follows:

- Our method does not require the material's structure, which is not always readily available, allowing for an efficient scanning of many different materials
- We show that tensor completion provides better performance for composition-based prediction with limited training data when compared with other baseline ML methods

- We conduct a thorough experimentation with various settings, including material property prediction with limited training data and fractional compounds
- We use tensor modeling to predict multiple material properties simultaneously, by introducing an additional tensor mode corresponding to the multiple properties

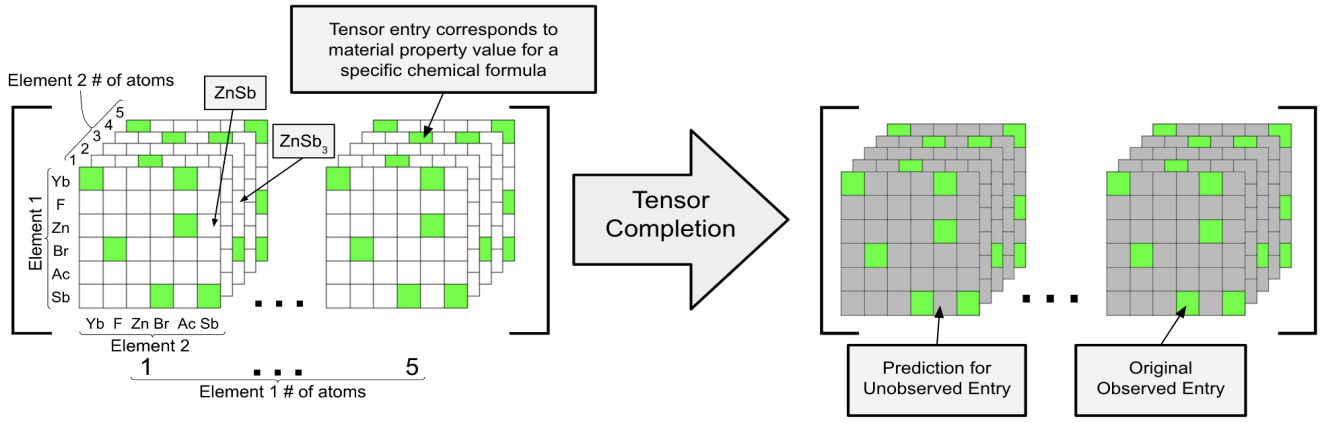


Figure 1. In this work we model several composition-based material property prediction tasks as instances of tensor completion. This allows us to take advantage of the structure in our tensor datasets to infer material property values for a vast number of combinations of materials. We explore different material property values to predict, as well as different variations of inputs (e.g. limited data, fractional compounds), to suggest that tensor completion is useful in a variety of settings.

2 Preliminaries

Here we outline some of the preliminaries and terminology used in this paper.

2.1 Tensors

Tensors are multidimensional arrays. In other words, a vector is a 1-dimensional tensor, and a matrix is a 2-dimensional tensor. We will be looking at tensors of 3 or more dimensions[31,32]. The tensors for our application will also be sparse tensors, which are tensors with many missing values. We use boldface italicized letters (e.g. $\mathcal{X}$) to denote dense tensors (tensors with no missing values). For sparse tensors, we will use the same notation, with a subscript "S". For example, $\mathcal{X}_S$ would be a sparse tensor corresponding to dense tensor $\mathcal{X}$.

2.1.1 Tensor Decomposition

Tensor decomposition is the process of expressing a tensor using smaller factors. For example, a common method of tensor decomposition is the Canonical Polyadic Decomposition (CPD)[31,32]. CPD expresses a tensor as a sum of rank-one tensors. A third-order tensor $\mathcal{X} \in \mathbb{R}^{I \mathbf{x} J \mathbf{x} K}$ would be expressed as: $\mathcal{X} \approx \sum_{r=1}^{R} (\mathbf{a}_r \circ \mathbf{b}_r \circ \mathbf{c}_r)$, where $\circ$ denotes outer product, $\mathbf{a}_r \in \mathbb{R}^I, \mathbf{b}_r \in \mathbb{R}^J$, and $\mathbf{c}_r \in \mathbb{R}^K$. We represent a decomposition of tensor $\mathcal{X}$ as $\mathcal{X}_D$.

2.1.2 Tensor Completion

Tensor Completion is the process of filling in the missing values of a sparse tensor, typically using on tensor decomposition. There are several forms of Tensor Completion methods. Commonly used tensor decomposition methods are Classical Tensor Methods such as CPD[33] & TuckER[34]. Recently Neural Tensor Methods have also become popular to leverage non-linear patterns (via neural networks) for tensor completion, such as CoSTCo[35] & NeAT[36].

Classical Tensor Methods heavily rely upon the tensor decomposition in order to infer back the missing values. CPD, for example, uses a series of matrix multiplication and addition in order to generate the dense tensor again. Neural Tensor Methods use an additional neural network to infer back the missing values of the tensor. CoSTCo[35], for example, uses a Convolutional Neural Network (CNN) to extract information from very sparse tensors to accurately infer back the missing values.

3 Methods

Here we describe the methods used in our experiments. More specifically, we describe our methods for the dataset generation as well as for tensor completion for materials' property prediction.

3.1 Original Datasets

For our experiments, we used 3 publicly available datasets. One from Li et al.[6], which contains the material property values band gap, formation energy per atom, energy above hull, and fermi energy. One dataset is from Hu et al.[7] which contains band gap information and includes fractional compounds. The last dataset is the HuggingFace LeMaterial dataset, from Martin Siron et al.[37–40], which contains total magnetization information.

3.2 Tensor Dataset Generation

To perform our tensor completion experiments, we first generate tensors to represent our material property prediction tasks, using the original datasets. For our tensors, generally each mode of the tensor would correspond to a different input variable, where the indices of that mode represent the values of that input variable. The entries of the tensor would correspond to the output variable. For example, a tensor in our experiment could have tensor modes to represent the elements in the chemical formula, and the tensor entries would be the material property value we want to predict, band gap for example.

For our composition-based material property prediction tasks, we generate tensors where the modes correspond to either the unique elements in the chemical formula or the amount of each element in the material. An example fourth-order tensor is shown in Figure 2. The highlighted value represents the chemical formula $AuBr_5$.

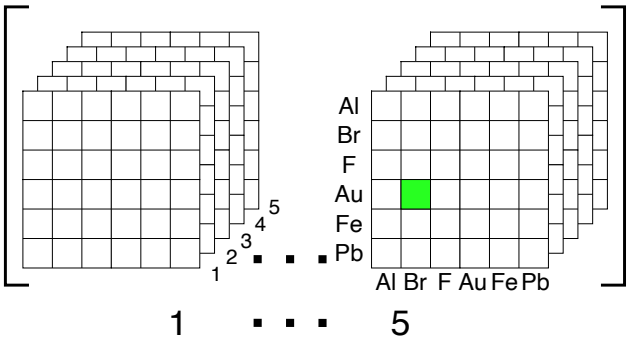


Figure 2. Example fourth order tensor representing materials' elements and elements' ratios. The highlighted entry could correspond to the chemical formula $AuBr_5$, since it corresponds to indices Au & Br for the elements, and indices 1 & 5 for the number of atoms in the material.

This example fourth-order tensor corresponds to a chemical formula with two unique elements. For our experiments, we also generate tensors of higher orders to represent chemical formulas with more elements. To ensure our tensors do not get exponentially large, we truncate the indices corresponding to rarely observed elements. For example, if we observe some of the elements very rarely in the entire dataset, we would remove the indices corresponding to those elements. We also would remove chemical formulas that include elements with very large number of atoms, since the indices associated with large (e.g. greater than 25) numbers of atoms are very rare in our datasets. This is done with the intention of limiting the number of indices in each tensor mode so that we can still derive useful information when we perform tensor decomposition.

3.2.1 Fractional Compounds

We also see several instances of fractional chemical formulas in our datasets (e.g. $Fe_{0.3}S$). To handle this type of data, we can simply consider the decimals as their own index of the tensor. With $Fe_{0.3}S$, we might have tensor indices corresponding to values [0.3, 0.5, 1, 1.5, 2...].

3.2.2 Redundancy Control

In our material property datasets, we observe significant amounts of redundant materials. For example, one chemical formula might appear in the same dataset multiple times, which could cause misleading evaluations if the same chemical formula is in both the training and evaluation data. For this reason, we remove all the duplicate chemical formulas from our datasets and then proceed. Redundancy in materials datasets has been cited as an issue for ML training, causing misleading evaluations[6,27,41].

By setting up our material property prediction tasks as tensors, we automatically remove all redundancies in practice (in terms of repeated chemical formulas), because one material can only correspond to exactly one tensor entry.

3.3 Tensor Completion for Materials Design

By using tensor completion, we leverage the inherent structure in our multidimensional datasets to predict the results of the entire space of combinations.

3.3.1 Tensor Completion Models

In our experiments, we use the following tensor completion models: **CPD**[33], **CPD-S**[42,43], **NeAT**[36], as well as **ensemble** tensor completion methods[42] to aggregate the results from multiple individual tensor completion models.

3.3.2 Tensor Completion Training

Training our tensor completion algorithms begins with randomly initializing a decomposition of the dense tensor $\mathscr{X}$. Using this random decomposition $\mathscr{X}_D$, we can generate an estimation of the dense tensor $\mathscr{X}$ according to the corresponding method's algorithm. For example, CPD uses matrix products and sums to reconstruct the full tensor, while CoSTCo[35] uses CNNs. After building the estimation of the dense tensor $\mathscr{X}$, we calculate the loss using the Mean Squared Error (MSE): $Loss = \text{Mean}[(\mathscr{X}_S - R(\mathscr{X}_D))^2 \cdot M_{\mathscr{X}}]$ where $R(\mathscr{X}_\mathscr{D})$ represents the full reconstruction of tensor $\mathscr{X}$ using only the randomly initialized factors $\mathscr{X}_D$. $M_{\mathscr{X}}$ is a boolean mask, whose values are 1 if that entry in $\mathscr{X}_S$ is observed, and 0 if it is unobserved. This is because we only keep the reconstruction for the observed indices of the sparse tensor to calculate our loss. We multiply the loss by the mask, to convert the loss for the missing entries to 0, and the non-missing entries remain the same. Now we can optimize the randomly initialized decomposition $\mathscr{X}_D$ in terms of MSE, using Adam[44] with backpropagation. We are essentially iteratively getting closer to a full reconstruction of $\mathscr{X}$ such that the entries corresponding to the observed values are similar to their actual values. We provide a visualization of a common tensor completion method, using CPD, in Figure 3.

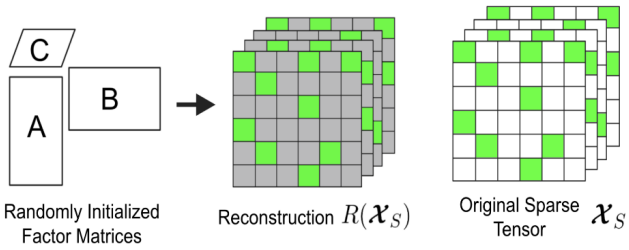


Figure 3. Visualization of CPD tensor completion training for a third-order tensor, for example. After factor matrices are randomly initialized, they are iteratively adjusted such that their produced reconstruction has values that correspond to the observed tensor values (highlighted in green). After convergence (the green values match each other sufficiently), the dark gray values are then used to infer the missing tensor values (the white cells of the original sparse tensor).

In all but one of our experiments, we randomly sample from the entire tensor for our training entries. The only exception is when we predict multiple properties at the same time, and we talk about how we sample training entries in Section 3.3.4.

3.3.3 Ensemble tensor completion training

Beyond the individual tensor completion methods we use, we also use an ensemble tensor completion method[42] to aggregate several individual tensor completion results. Of the various aggregation functions we tried (mean, median, MLPs, CNNs, etc.), we found that Sklearn's[45] Histogram Gradient Boosting gave the best performance, so this is what we use in our experiments.

In Figure 4 we use multiple rank decompositions to complete the same tensor, since the true rank of a sparse tensor will be unknown. We use this method in the absence of an optimal estimate of the true rank of the tensor. In addition to this, we train each individual tensor completion algorithm on a slightly different piece of the same sparse tensor to extract different patterns for tensor completion. For example, each individual tensor completion algorithm might train on a different 80% of the sparse data. There will be overlapping training data, but the entire set of training data is unique between the individual algorithms.

3.3.4 Multiple Property Prediction

To leverage the capabilities of tensor modeling, we also predict multiple material properties simultaneously, by introducing a new tensor mode which corresponds to the different material properties to predict. In our experiments, we discovered that simply randomly sampling the entire tensor for training entries is not as effective as randomly sampling unique materials and then using all material properties from those materials as training values.

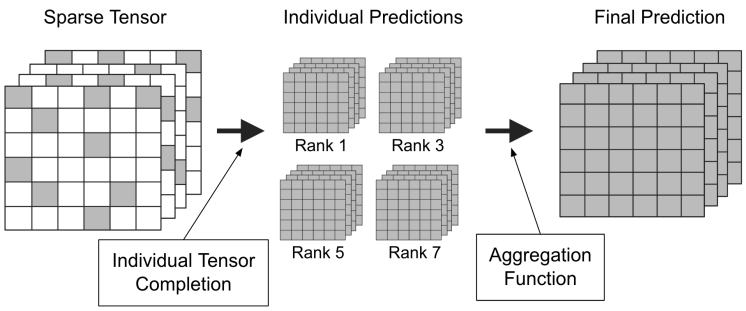


Figure 4. Visualization of the ensemble tensor completion, using four individual tensor completion methods based on rank 1, 3, 5, and 7 decompositions. These individual predictions are aggregated to form one completed tensor.

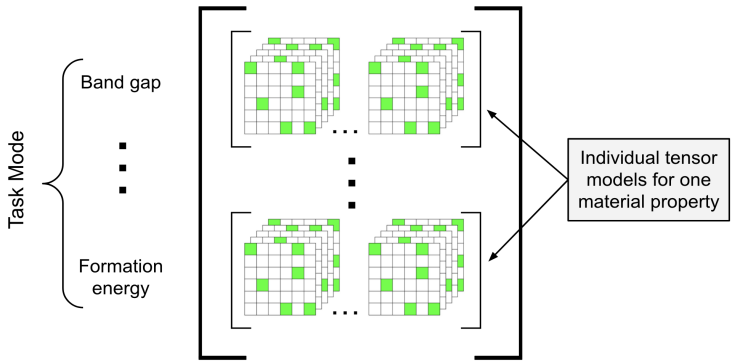


Figure 5. Visualization our task mode, to model multiple material property prediction tasks as a single higher-order tensor.

3.4 Evaluation

Here we outline our methods of evaluation our tensor completion methods for these tasks, including evaluation metric and baseline comparisons.

3.4.1 Evaluation Metric

From the observed sparse tensor values, we randomly sample the training values and use the rest of the values to calculate the mean absolute error (MAE). For a true value y_i and a predicted value $\hat{y}_i$, we calculate it as $MAE = \frac{1}{n}\sum_{i=1}^{n}|y_i - \hat{y}_i|$. One important note is that all the values are normalized (using standard normalization), before the experiments are performed. The normalization only uses the training data statistics (i.e. all data is normalized using mean & standard deviation of just the training data).

3.4.2 Comparison Against Baseline ML Models

We compare the performance of our tensor completion models with a few baseline models. We use Sklearn's HistGradientBoostingRegressor (HGB) implementation[45], and we use the XGBoost (XGB) implementation[30]. We also implement a Multi-Layer Perceptron (MLP) & Graph Neural Network (GNN) using PyTorch[46]. Typically, GNNs are very useful for structure-based material property prediction tasks; however, since our goal is to use only the chemical formula, we model our GNN implementation similarly to Xue et al.[24] to be structure-agnostic. Each unique element corresponds to a node with its number of atoms and type of element. We treat each chemical formula as its own fully connected graph. When training and evaluating our baseline methods, we use samples from original tabular datasets, corresponding to the same training and testing samples in our generated tensors.

4 Experimental Evaluation

Here we experimentally evaluate the viability of using tensor completion in material property prediction tasks, across a wide variety of scenarios. We empirically show the utility of tensor completion for different composition-based material property prediction tasks.

4.1 General Model Comparison

For each prediction method, we display the average MAE over 5 iterations. We also compare non-tensor ML models: HGB[45], XGBoost[30], MLP[46], and GNN[24,46]. For our GNN implementation, we set up each material as its own fully-connected graph, where the nodes represent a unique element of the material. Each node has features corresponding to the element & number of atoms, all the information from the chemical formula.

	Formation Energy			Energy above Hull			Fermi Energy		
Model \ # Elements	2	3	4	2	3	4	2	3	4
CPD	0.63 ± 0.1	0.48 ± 0.0	0.67 ± 0.1	0.55 ± 0.1	0.54 ± 0.0	0.49 ± 0.1	0.70 ± 0.0	0.66 ± 0.1	0.76 ± 0.0
CPD-S	0.62 ± 0.1	0.46 ± 0.0	0.54 ± 0.1	0.54 ± 0.1	0.51 ± 0.0	0.45 ± 0.1	0.68 ± 0.0	0.65 ± 0.1	0.68 ± 0.0
NeAT	0.58 ± 0.1	0.44 ± 0.1	0.49 ± 0.1	$\mathbf{0.50} \pm 0.1$	$\mathbf{0.47} \pm 0.1$	$\mathbf{0.44} \pm 0.1$	0.64 ± 0.0	0.57 ± 0.0	0.58 ± 0.0
NeAT Ensemble	$\mathbf{0.44} \pm 0.0$	$\mathbf{0.36} \pm 0.0$	$\mathbf{0.36} \pm 0.0$	0.54 ± 0.1	0.51 ± 0.0	0.51 ± 0.1	$\mathbf{0.52} \pm 0.0$	$\mathbf{0.51} \pm 0.0$	$\mathbf{0.51} \pm 0.0$
Baselines									
HGB	0.57 ± 0.0	0.38 ± 0.0	0.44 ± 0.0	0.51 ± 0.1	0.52 ± 0.0	0.52 ± 0.1	0.72 ± 0.0	0.58 ± 0.0	0.56 ± 0.0
XGB	0.59 ± 0.0	0.56 ± 0.0	0.69 ± 0.1	$\mathbf{0.50} \pm 0.1$	0.50 ± 0.0	0.51 ± 0.1	0.64 ± 0.0	0.66 ± 0.0	0.70 ± 0.0
MLP	0.68 ± 0.0	0.62 ± 0.0	0.73 ± 0.1	0.55 ± 0.1	0.51 ± 0.0	0.51 ± 0.1	0.76 ± 0.0	0.75 ± 0.0	0.73 ± 0.0
GNN	0.69 ± 0.1	0.71 ± 0.1	0.83 ± 0.1	$\mathbf{0.50} \pm 0.1$	0.52 ± 0.1	0.51 ± 0.1	0.75 ± 0.0	0.78 ± 0.0	0.79 ± 0.0

	Band Gap			Total Magnetization		
Model \ # Elements	2	3	4	2	3	4
CPD	0.57 ± 0.0	0.67 ± 0.1	0.83 ± 0.0	0.62 ± 0.1	0.60 ± 0.1	0.66 ± 0.1
CPD-S	0.57 ± 0.0	0.62 ± 0.1	0.83 ± 0.0	0.63 ± 0.1	0.56 ± 0.1	0.65 ± 0.1
NeAT	0.55 ± 0.1	0.51 ± 0.0	0.75 ± 0.0	0.61 ± 0.1	0.51 ± 0.1	0.56 ± 0.1
NeAT Ensemble	$\mathbf{0.42} \pm 0.0$	$\mathbf{0.45} \pm 0.0$	$\mathbf{0.68} \pm 0.0$	$\mathbf{0.46} \pm 0.1$	$\mathbf{0.42} \pm 0.1$	$\mathbf{0.51} \pm 0.0$
Baselines						
HGB	0.52 ± 0.0	0.48 ± 0.0	0.73 ± 0.0	0.61 ± 0.1	0.62 ± 0.1	0.66 ± 0.0
XGB	0.46 ± 0.0	0.56 ± 0.0	0.76 ± 0.0	0.53 ± 0.1	0.50 ± 0.1	0.57 ± 0.1
MLP	0.57 ± 0.1	0.61 ± 0.1	0.81 ± 0.0	0.59 ± 0.1	0.58 ± 0.0	0.64 ± 0.0
GNN	0.49 ± 0.1	0.63 ± 0.1	0.82 ± 0.0	0.61 ± 0.1	0.59 ± 0.1	0.63 ± 0.0

Table 1. Here we calculate the average & standard deviation of the MAE over 5 iterations using only 300 training values, for five different material properties to predict. These properties are band gap, formation energy, energy above hull, fermi energy, and total magnetization. The best results for each task (column) are displayed in bold font.

From our results in Table 1, we observe that for a variety of tasks, tensor completion, especially the NeAT Ensemble[36], has the best performance when compared against our baseline methods.

4.2 Band Gap Prediction

In this section we take a closer look at band gap prediction in particular, because tensor completion seems to work particularly well for predicting this material property. In Table 2 we look at band gap prediction performance, for a variety of scenarios. We observe band gap prediction performance using a dataset with only whole compounds[6], as well as a dataset with fractional compounds[7]. For both of these datasets, we test performance for chemical formulas with 2, 3, and 4 unique elements as input.

From these results in Table 2, we observe that tensor completion models, especially NeAT[36], usually exhibit superior performance in band gap prediction tasks, across a variety of scenarios. This suggests that modeling band gap prediction as a tensor completion task is especially beneficial, as opposed to a standard supervised ML task.

4.2.1 Disaggregated Performance

For these band gap prediction tasks, we also disaggregate the performance, to take a closer look at inference behavior. More specifically, we observe performance for when band gap is 0 (conductors) and when band gap is greater than 0 (semiconductors & insulators) separately. This is similar to the classification metrics precision and recall, where conductors is the majority class in our datasets, and semiconductors & insulators are the minority.

From the results in Table 3, we see that tensor completion, more specifically the NeAT Ensemble, usually offers the best performance for both the conductors, as well as the semiconductors & insulators.

4.3 Data Efficiency Analysis

Here we observe the performance for different models with respect to the number of training values, to see how many training entries we need to accurate complete the tensors.

	Dataset 1[6]			Dataset 2[7]		
# Elements	2	3	4	2	3	4
CPD	0.57 ± 0.0	0.66 ± 0.1	0.83 ± 0.0	0.68 ± 0.1	0.73 ± 0.0	0.70 ± 0.0
CPD-S	0.57 ± 0.0	0.62 ± 0.1	0.83 ± 0.0	0.67 ± 0.1	0.72 ± 0.0	0.68 ± 0.1
NeAT	0.49 ± 0.1	0.56 ± 0.0	0.72 ± 0.0	0.61 ± 0.1	0.53 ± 0.0	0.61 ± 0.0
NeAT Ensemble	**0.42** ± 0.0	**0.45** ± 0.0	**0.68** ± 0.0	**0.42** ± 0.1	**0.41** ± 0.0	**0.50** ± 0.0
Baselines						
HGB	0.52 ± 0.0	0.48 ± 0.0	0.73 ± 0.0	0.52 ± 0.0	0.45 ± 0.0	0.55 ± 0.0
XGB	0.46 ± 0.0	0.56 ± 0.0	0.76 ± 0.0	0.58 ± 0.1	0.61 ± 0.0	0.65 ± 0.0
MLP	0.54 ± 0.0	0.60 ± 0.1	0.81 ± 0.0	0.62 ± 0.1	0.68 ± 0.0	0.80 ± 0.0
GNN	0.60 ± 0.1	0.69 ± 0.1	0.83 ± 0.0	0.70 ± 0.1	0.77 ± 0.0	0.79 ± 0.0

Table 2. Comparison of ML models for band gap prediction, using the average & standard deviation of the MAE over 5 iterations. The best result for each task (column) is displayed in bold. Dataset 2[7] also includes fractional compounds, while Dataset 1[6] does not.

Conductors

	Dataset 1[6]			Dataset 2[7]		
Model \ # Elements	2	3	4	2	3	4
CPD	0.35 ± 0.0	0.46 ± 0.1	0.87 ± 0.0	0.56 ± 0.0	0.70 ± 0.0	1.26 ± 0.1
CPD-S	0.35 ± 0.1	0.40 ± 0.1	0.85 ± 0.0	0.56 ± 0.0	0.69 ± 0.0	1.29 ± 0.2
NeAT	0.26 ± 0.1	0.36 ± 0.1	**0.56** ± 0.2	0.48 ± 0.2	0.39 ± 0.1	0.99 ± 0.2
NeAT Ensemble	**0.21** ± 0.0	**0.22** ± 0.0	0.57 ± 0.1	**0.23** ± 0.0	**0.21** ± 0.0	**0.67** ± 0.1
Baselines						
HGB	0.31 ± 0.0	0.30 ± 0.0	0.67 ± 0.0	0.36 ± 0.0	0.35 ± 0.0	0.77 ± 0.1
XGB	0.24 ± 0.0	0.34 ± 0.0	0.75 ± 0.0	0.40 ± 0.0	0.49 ± 0.0	1.12 ± 0.0
MLP	0.33 ± 0.1	0.39 ± 0.1	0.76 ± 0.0	0.40 ± 0.0	0.52 ± 0.0	1.31 ± 0.1
GNN	0.40 ± 0.1	0.48 ± 0.1	0.83 ± 0.0	0.55 ± 0.1	0.63 ± 0.0	1.33 ± 0.1

Semiconductors & Insulators

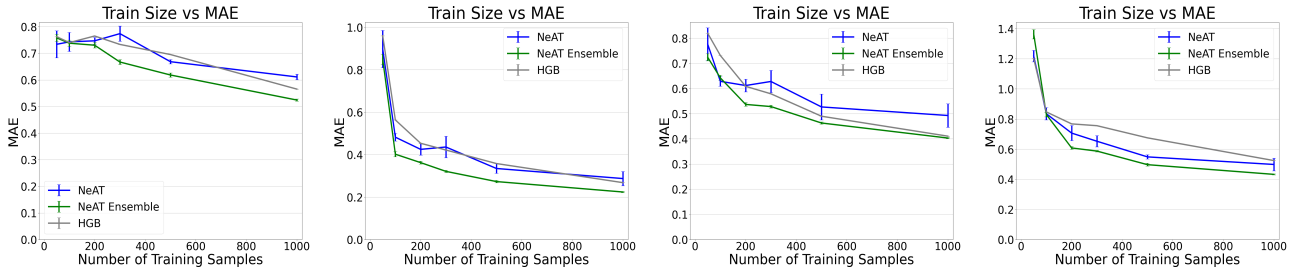
	Dataset 1[6]			Dataset 2[7]		
Model \ # Elements	2	3	4	2	3	4
CPD	1.69 ± 0.1	1.35 ± 0.1	0.81 ± 0.0	0.87 ± 0.1	0.77 ± 0.0	0.58 ± 0.0
CPD-S	1.69 ± 0.1	1.33 ± 0.1	0.81 ± 0.0	0.85 ± 0.1	0.76 ± 0.1	0.54 ± 0.1
NeAT	1.71 ± 0.2	1.23 ± 0.1	0.82 ± 0.1	0.81 ± 0.1	0.69 ± 0.1	0.53 ± 0.0
NeAT Ensemble	**1.49** ± 0.1	1.21 ± 0.1	**0.74** ± 0.0	**0.72** ± 0.1	0.64 ± 0.0	**0.46** ± 0.1
Baselines						
HGB	1.57 ± 0.1	**1.08** ± 0.1	0.76 ± 0.0	0.76 ± 0.1	**0.56** ± 0.0	0.51 ± 0.0
XGB	1.62 ± 0.1	1.33 ± 0.1	0.78 ± 0.0	0.86 ± 0.1	0.73 ± 0.1	0.54 ± 0.0
MLP	1.65 ± 0.1	1.32 ± 0.1	0.84 ± 0.0	0.96 ± 0.2	0.86 ± 0.1	0.69 ± 0.1
GNN	1.64 ± 0.1	1.37 ± 0.1	0.84 ± 0.0	0.93 ± 0.1	0.93 ± 0.1	0.67 ± 0.1

Table 3. Comparison of ML models for band gap prediction, using the disaggregated MAE results. Samples that are conductors (band gap = 0) are evaluated independently of semiconductors & insulators (band gap > 0). Average MAE over 5 iterations is taken, and the best result for each task (column) is in bold. Dataset 2[7] includes fractional compounds, while Dataset 1[6] does not.

From Figure 6, we can see that for a variety of tasks, the NeAT Ensemble[43] requires less training values than the other tensor completion methods. This makes the NeAT Ensemble especially useful when the material property values of materials are expensive or time-consuming to compute, since we need less training values to accurately predict unobserved materials' property values.

4.4 Individual Samples' Comparison

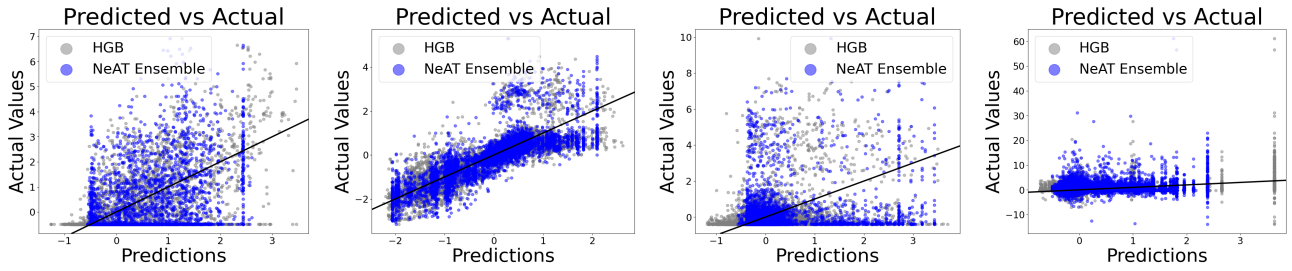
We also look at the individual predictions versus the true values, to learn more about the models' performances than what the aggregate errors tell us. In Figure 7, we plot the true value on the y-axis and the predicted value on the x-axis for each model. A



(a) Band gap prediction (b) Formation energy per atom prediction (c) Fermi energy prediction (d) Total magnetization prediction

Figure 6. Training size versus the MAE. The MAE is calculated over 5 iterations, and the average and standard deviation are displayed.

perfect model (MAE = 0) would have all the predictions lying exactly on the black diagonal line.



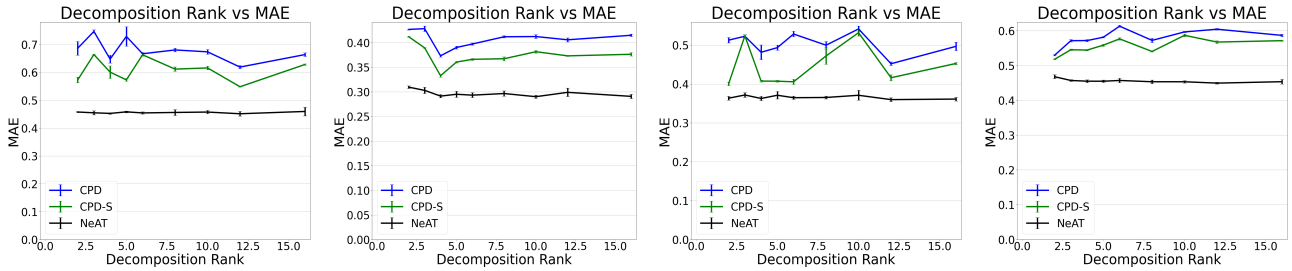
(a) Band gap prediction (b) Formation energy prediction (c) Energy above hull prediction (d) Total magnetization prediction

Figure 7. Individual predicted versus actual values. Tensor models in blue & green, non-tensor models in gray & red. A perfect model's predictions would lie on the black line. We use NeAT Ensemble as our tensor completion model, and HGB for our baseline comparison model.

In Figure 7 we get a closer look at the individual predictions of our models. We observe that tasks such as band gap and energy above hull prediction are significantly harder to generate meaningful predictions, likely because the majority of data having a true value of 0. For formation energy and total magnetization prediction, on the other hand, we see that we are able to generate more useful predictions.

4.5 Approximation Rank vs MAE Plots

One crucial hyperparameter in our tensor completion algorithms is the decomposition rank we choose. For this reason, we conduct experiments to observe the performance with respect to different decomposition ranks used in the tensor completion.



(a) Band gap prediction (b) Formation energy prediction (c) Energy above hull prediction (d) Fermi energy

Figure 8. Tensor completion performance for various decomposition ranks. The MAE is taken over 5 iterations, and the mean and standard deviation are displayed. Training size is fixed at 2500 samples.

From Figure 8, we observe that both CPD and CPD-S are highly volatile with respect to different decomposition ranks. NeAT, on the other hand, shows impressive reliability for different decomposition ranks. NeAT's reliability is very crucial since it is impossible to know the true rank of the full tensor when we are trying to complete highly sparse tensors in practical applications.

4.6 Data Augmentation

One issue that often arises with ML is a lack of training data, especially in practice with real world datasets. To maintain performance with increasingly sparse datasets, we consider methods of data augmentation. The data augmentation methods we use are as follows:

1. Naive method – randomly oversampling the training set (i.e. intentionally creating duplicates only in the train set)
2. Using the DAIN[47] data augmentation technique to generate synthetic data
3. Making predictions using NeAT[36], and then randomly sampling these predictions as additional training values for a different tensor completion model

4.6.1 General Comparison

First we conduct experiments across a variety of material properties to predict with different numbers of unique elements in the chemical formula. We compare our previously mentioned data augmentation techniques with the original training set (i.e. no data augmentation).

	Formation Energy			Energy above Hull			Fermi Energy		
Aug. \ # Elements	2	3	4	2	3	4	2	3	4
None	0.66 ± 0.1	0.37 ± 0.1	1.58 ± 0.7	0.13 ± 0.0	0.17 ± 0.0	0.10 ± 0.0	1.25 ± 0.0	1.09 ± 0.0	1.21 ± 0.0
Random	0.64 ± 0.1	0.39 ± 0.1	**1.33** ± 0.6	0.14 ± 0.0	0.18 ± 0.0	0.10 ± 0.0	1.30 ± 0.0	1.12 ± 0.0	1.26 ± 0.0
DAIN	**0.34** ± 0.0	0.29 ± 0.0	1.58 ± 0.7	**0.13** ± 0.0	**0.16** ± 0.0	**0.10** ± 0.0	1.21 ± 0.0	1.06 ± 0.0	1.14 ± 0.0
NeAT	0.35 ± 0.0	**0.28** ± 0.0	1.49 ± 0.6	0.14 ± 0.0	0.16 ± 0.0	0.10 ± 0.0	**1.19** ± 0.0	**1.06** ± 0.0	**1.13** ± 0.0

	Band Gap			Total Magnetization		
Aug. \ # Elements	2	3	4	2	3	4
None	**0.28** ± 0.0	0.30 ± 0.0	0.93 ± 0.0	1.56 ± 0.0	**0.92** ± 0.0	1.25 ± 0.0
Random	0.28 ± 0.0	0.31 ± 0.0	0.94 ± 0.0	1.60 ± 0.0	0.94 ± 0.0	1.30 ± 0.0
DAIN	0.28 ± 0.0	**0.28** ± 0.0	**0.87** ± 0.0	**1.48** ± 0.0	0.98 ± 0.0	1.22 ± 0.0
NeAT	0.29 ± 0.0	0.29 ± 0.0	0.87 ± 0.0	1.50 ± 0.0	0.94 ± 0.0	**1.21** ± 0.0

Table 4. Here we compare various data augmentation (shortened to aug. in table) methods for a variety of tasks. For these experiments, there are 1000 original training samples, and 500 augmented samples for a total of 1500 training samples. When augmentation is 'None', only the 1000 original training values are used. We calculate the average & standard deviation of the MAE over 5 iterations. The best results for each task (column) are displayed in bold font.

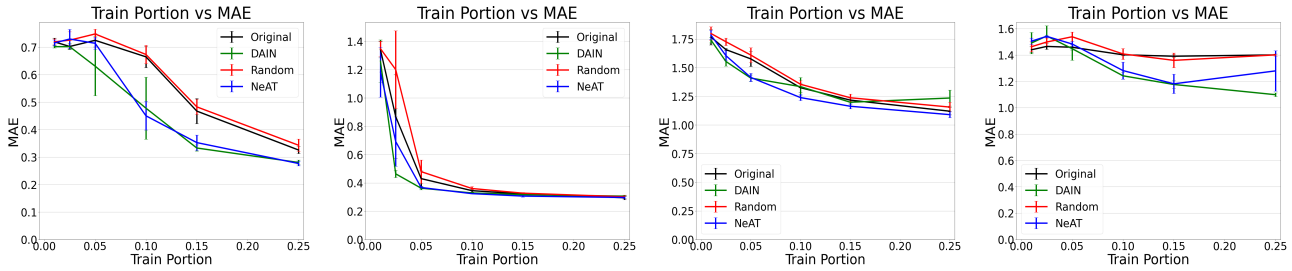
From Table 4 we observe that data augmentation is really only useful for a few of the material properties to predict. However, in cases when it is not necessarily useful, data augmentation does not seem to hinder performance. As expected, randomly oversampling the training samples (naive method) does not really do anything performance-wise. On the other hand, using DAIN[47] and the NeAT-based[36] data augmentation techniques do improve performance when predicting formation energy, Fermi energy, and total magnetization.

4.6.2 Augmented Dataset for Different Training Sizes

Here we compare tensor completion performance using the original dataset with different data augmentation techniques, for a different amounts of training data. For each training size, calculate MAE on the remainder of the data (which was not trained on).

Our results in figure 9 display how data augmentation can benefit our material property prediction tasks for various training sizes. In these experiments, if the original data's train set contains n samples, the augmented data's train set will have those original n samples, plus an additional $\frac{n}{2}$ synthetic samples, so $\frac{3n}{2}$ total samples.

When predicting total magnetization and formation energy per atom, it seems that some sophisticated form of data augmentation can be beneficial, whether we use NeAT[36] or DAIN[47].

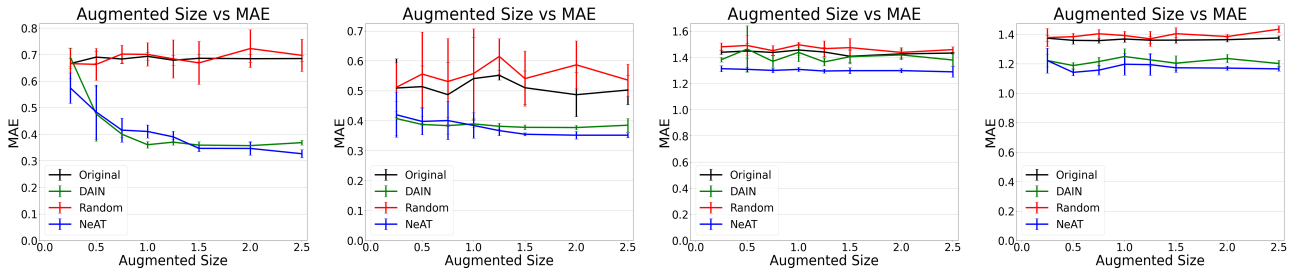


(a) Formation energy per atom prediction for 2 elements (b) Formation energy per atom prediction for 3 elements (c) Total magnetization prediction for 2 elements (d) Total magnetization prediction for 3 elements

Figure 9. Average and standard deviation of MAE over 5 iterations, using CPD tensor completion. We compare training on only the original tensor dataset with several data augmentation techniques for additional training data. The x-axis represents the proportion of observed entries used for training.

4.6.3 How Many Augmented Samples?

We also conduct experiments to observe what amount of augmented samples are the most beneficial. In these experiments, we fix the training size, and we test a variety of amounts of augmented training samples. We consider the amount of augmented training samples as a proportion of the number of training entries.



(a) Formation energy per atom prediction for 2 elements (b) Formation energy per atom prediction for 3 elements (c) Total magnetization prediction for 2 elements (d) Total magnetization prediction for 3 elements

Figure 10. Average & standard deviation of MAE over 5 iterations for CPD tensor completion. We compare training on only the original tensor dataset with several data augmentation techniques for additional training data. The x-axis represents the amount of augmented samples, as a proportion of the original training size. Plots A and B use 1000 training entries. Plots C and D use 5000 training entries.

4.7 Predicting Multiple Material Properties

For our dataset from Li et al.[6], we have multiple material properties for each chemical formula: (i) band gap (ii) formation energy per atom (iii) energy above hull and (iv) Fermi energy. To leverage the capabilities of tensors, we create an additional tensor mode corresponding to the different material properties so we can predict all 4 material properties simultaneously.

In these experiments, we evaluate the performance of tensor completion with and without a task mode. After performing some sampling experiments, we determined that randomly sampling the entire tensor for training entries is not that effective. For this reason, we randomly sample the chemical formulas, and use all 4 material properties from those randomly sampled chemical formulas for tensor completion training.

From Table 5, we observe that adding a task mode, or predicting all four material properties at once, usually does not do much for performance. In some cases, however, it does improve performance, like with band gap prediction of 2 & 3 unique element compounds. It is important to note that even though the additional task mode does not help for all tasks, we do not really see any significant adverse affects.

5 Conclusion

We show the utility of composition-based material property prediction, which can be especially useful for materials design when the material's structure is difficult or expensive to calculate. We model these composition-based material property prediction

	Band Gap			Formation Energy		
Method \ # Elements	2	3	4	2	3	4
No Task Mode	0.60 ± 0.1	0.53 ± 0.1	$\mathbf{1.12} \pm 0.0$	$\mathbf{0.52} \pm 0.0$	0.42 ± 0.0	$\mathbf{0.46} \pm 0.0$
Task Mode	$\mathbf{0.38} \pm 0.0$	$\mathbf{0.42} \pm 0.0$	1.13 ± 0.0	0.55 ± 0.0	$\mathbf{0.40} \pm 0.0$	0.49 ± 0.0

	Energy Above Hull			Fermi Energy		
Method \ # Elements	2	3	4	2	3	4
No Task Mode	$\mathbf{0.21} \pm 0.0$	0.25 ± 0.0	0.15 ± 0.0	$\mathbf{1.98} \pm 0.1$	$\mathbf{1.49} \pm 0.1$	1.50 ± 0.1
Task Mode	0.23 ± 0.0	$\mathbf{0.25} \pm 0.0$	$\mathbf{0.13} \pm 0.0$	1.99 ± 0.1	1.59 ± 0.1	$\mathbf{1.49} \pm 0.1$

Table 5. Comparison of predicting material properties individually (no task mode) and when combining all individual tensors into one tensor with a task mode. Average & MAE is taken over 5 iterations, using 100 unique materials for training. Without a task mode, this means 100 training entries. With the task mode, this is 100 materials $\times$ 4 material properties = 400 training entries. In this experiment, the data is not normalized.

tasks as tensor completion tasks, and we are able to more accurately infer the properties of unobserved materials. We show that tensor methods exhibit better performance in these composition-based material property prediction when compared with baseline ML methods, especially when dealing with limited training data. Furthermore, we evaluate the usefulness of data augmentation techniques, and predicting multiple material properties simultaneously.

A limitation of our approach is that tensor completion will generate many predictions for materials that cannot exist. This is because we will complete the entire tensor of material combinations, which will most likely include compounds that are infeasible. For now we defer this issue to domain-experts, who are able to determine the feasibility of various materials. However, we hope to address this in a follow-up work by predicting both the material property value and the feasibility of the chemical formula. In principle, "feasibility" would be an additional mode in our tensor, much like our multi-task prediction scheme, assuming we had this data produced by material synthesis experiments.

Data Availability

We used publicly available datasets in these experiments. Please refer to Section 3.1 Original Datasets for more information.

References

1. Hafner, J., Wolverton, C. & Ceder, G. Toward computational materials design: the impact of density functional theory on materials research. *MRS bulletin* **31**, 659–668 (2006).
2. Verma, P. & Truhlar, D. G. Status and challenges of density functional theory. *Trends Chem.* **2**, 302–318 (2020).
3. Schleder, G. R., Padilha, A. C., Acosta, C. M., Costa, M. & Fazzio, A. From dft to machine learning: recent approaches to materials science–a review. *J. Physics: Mater.* **2**, 032001 (2019).
4. Makkar, P. & Ghosh, N. N. A review on the use of dft for the prediction of the properties of nanomaterials. *RSC advances* **11**, 27897–27924 (2021).
5. Gongora, A. E. *et al.* Accelerating the design of lattice structures using machine learning. *Sci. Reports* **14**, 13703 (2024).
6. Li, Q., Fu, N., Omee, S. S. & Hu, J. Md-hit: Machine learning for material property prediction with dataset redundancy control. *npj Comput. Mater.* **10**, 245 (2024).
7. Hu, J., Liu, D., Fu, N. & Dong, R. Realistic material property prediction using domain adaptation based machine learning. *Digit. Discov.* **3**, 300–312 (2024).
8. Haghshenas, Y. *et al.* Full prediction of band potentials in semiconductor materials. *Mater. Today Phys.* **46**, 101519 (2024).
9. Wang, Y. *et al.* Accelerated design of fe-based soft magnetic materials using machine learning and stochastic optimization. *Acta Materialia* **194**, 144–155 (2020).
10. Pakala, S., Ahn, D. & Papalexakis, E. Tensor completion for surrogate modeling of material property prediction. *arXiv preprint arXiv:2501.18137* (2025).
11. Alberi, K. *et al.* The 2019 materials by design roadmap. *J. Phys. D: Appl. Phys.* **52**, 013001 (2018).
12. Zhuo, Y., Mansouri Tehrani, A. & Brgoch, J. Predicting the band gaps of inorganic solids by machine learning. *The journal physical chemistry letters* **9**, 1668–1673 (2018).

13. Seko, A., Maekawa, T., Tsuda, K. & Tanaka, I. Machine learning with systematic density-functional theory calculations: Application to melting temperatures of single-and binary-component solids. *Phys. Rev. B* **89**, 054303 (2014).

14. Wang, T., Zhang, K., Thé, J. & Yu, H. Accurate prediction of band gap of materials using stacking machine learning model. *Comput. Mater. Sci.* **201**, 110899 (2022).

15. Wan, Z., Wang, Q.-D., Liu, D. & Liang, J. Effectively improving the accuracy of pbe functional in calculating the solid band gap via machine learning. *Comput. Mater. Sci.* **198**, 110699 (2021).

16. Li, X., Shan, G. & Shek, C. Machine learning prediction of magnetic properties of fe-based metallic glasses considering glass forming ability. *J. Mater. Sci. & Technol.* **103**, 113–120 (2022).

17. Jha, D. *et al.* Enhancing materials property prediction by leveraging computational and experimental data using deep transfer learning. *Nat. communications* **10**, 5316 (2019).

18. Xie, T. & Grossman, J. C. Crystal graph convolutional neural networks for an accurate and interpretable prediction of material properties. *Phys. review letters* **120**, 145301 (2018).

19. Jha, D., Gupta, V., Liao, W.-k., Choudhary, A. & Agrawal, A. Moving closer to experimental level materials property prediction using ai. *Sci. reports* **12**, 11953 (2022).

20. Mao, Y. *et al.* Prediction and classification of formation energies of binary compounds by machine learning: an approach without crystal structure information. *ACS omega* **6**, 14533–14541 (2021).

21. Dong, R., Fu, N., Siriwardane, E. M. & Hu, J. Generative design of inorganic compounds using deep diffusion language models. *The J. Phys. Chem. A* **128**, 5980–5989 (2024).

22. Dong, R., Song, Y., Siriwardane, E. M. & Hu, J. Discovery of 2d materials using transformer network-based generative design. *Adv. Intell. Syst.* **5**, 2300141 (2023).

23. Mortazavi, B. Recent advances in machine learning-assisted multiscale design of energy materials. *Adv. Energy Mater.* 2403876 (2024).

24. Xue, S.-D. & Hong, Q.-J. Materials properties prediction (mapp): Empowering the prediction of material properties solely based on chemical formulas. *Materials* **17**, 4176 (2024).

25. Huang, H., Magar, R. & Barati Farimani, A. Pretraining strategies for structure agnostic material property prediction. *J. Chem. Inf. Model.* **64**, 627–637 (2024).

26. Tian, S. I. P., Walsh, A., Ren, Z., Li, Q. & Buonassisi, T. What information is necessary and sufficient to predict materials properties using machine learning? (2022). 2206.04968.

27. Venkatraman, V. The utility of composition-based machine learning models for band gap prediction. *Comput. Mater. Sci.* **197**, 110637 (2021).

28. Sidiropoulos, N. D. *et al.* Tensor decomposition for signal processing and machine learning. *IEEE Transactions on signal processing* **65**, 3551–3582 (2017).

29. Scarselli, F., Gori, M., Tsoi, A. C., Hagenbuchner, M. & Monfardini, G. The graph neural network model. *IEEE transactions on neural networks* **20**, 61–80 (2008).

30. Chen, T. & Guestrin, C. Xgboost: A scalable tree boosting system. In *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD '16, 785–794, DOI: 10.1145/2939672.2939785 (ACM, 2016).

31. Kolda, T. & Bader, B. Tensor decompositions and applications. *SIAM review* **51** (2009).

32. Sidiropoulos, N. D. *et al.* Tensor decomposition for signal processing and machine learning. *IEEE Signal Process. Mag.* (2016).

33. Harshman, R. Foundations of the parafac procedure: Models and conditions for an" explanatory" multimodal factor analysis. *UCLA working papers phonetics* (1970).

34. Balažević, I., Allen, C. & Hospedales, T. M. Tucker: Tensor factorization for knowledge graph completion. In *Empirical Methods in Natural Language Processing* (2019).

35. Liu, H., Li, Y., Tsang, M. & Liu, Y. Costco: A neural tensor completion model for sparse tensors. In *Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, 324–334 (2019).

36. Ahn, D., Saini, U. S., Papalexakis, E. E. & Payani, A. Neural additive tensor decomposition for sparse tensors. In *33rd ACM International Conference on Information and Knowledge Management* (ACM, 2024).

37. Martin Siron *et al.* Lemat-bulkunique dataset (2024).

38. Jain, A. *et al.* Commentary: The materials project: A materials genome approach to accelerating materials innovation. *APL materials* **1** (2013).

39. Saal, J. E., Kirklin, S., Aykol, M., Meredig, B. & Wolverton, C. Materials design and discovery with high-throughput density functional theory: the open quantum materials database (oqmd). *Jom* **65**, 1501–1509 (2013).

40. Schmidt, J., Pettersson, L., Verdozzi, C., Botti, S. & Marques, M. A. Crystal graph attention networks for the prediction of stable materials. *Sci. advances* **7**, eabi7948 (2021).

41. Li, K. *et al.* Exploiting redundancy in large materials datasets for efficient machine learning with less data. *Nat. Commun.* **14**, 7283 (2023).

42. Pakala, S. *et al.* Automating data science pipelines with tensor completion. In *2024 IEEE International Conference on Big Data (BigData)*, 1075–1084, DOI: 10.1109/BigData62323.2024.10825934 (2024).

43. Ahn, D., Jang, J.-G. & Kang, U. Time-aware tensor decomposition for sparse tensors. In *2021 IEEE 8th International Conference on Data Science and Advanced Analytics (DSAA)*, 1–2 (IEEE, 2021).

44. Kingma, D. P. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980* (2014).

45. Pedregosa, F. *et al.* Scikit-learn: Machine learning in Python. *J. Mach. Learn. Res.* **12**, 2825–2830 (2011).

46. Paszke, A. *et al.* Pytorch: An imperative style, high-performance deep learning library. *Adv. neural information processing systems* **32** (2019).

47. Oh, S., Kim, S., Rossi, R. A. & Kumar, S. Influence-guided data augmentation for neural tensor completion. In *Proceedings of the 30th ACM International Conference on Information & Knowledge Management*, 1386–1395 (2021).

Acknowledgements

We would like to thank Dawon Ahn for her intial discussions on this work. Research was supported by the National Science Foundation CAREER grant no. IIS 2046086, grant no. IIS 1901379, and CREST Center for Multidisciplinary Research Excellence in Cyber-Physical Infrastructure Systems (MECIS) grant no. 2112650.

Author contributions statement

S.P. conducted the experiments, drafted the manuscript, and contributed to research ideas. E.P. provided research directions and ideas, along with revising the manuscript, supervising, and securing funding.

Competing interests

The authors declare no competing interests.