Dynamic Embedding Representation for Graph Neural Networks to Enhance Materials Property Prediction with Limited Datasets

Vishu Gupta[1,2,3], Kamal Choudhary[4], Youjia Li[1], Muhammed Nur Talha Kilic[5], Daniel Wines[4], Wei-keng Liao[1], Alok Choudhary[1], Ankit Agrawal[1]

[1]*Department of Electrical and Computer Engineering, Northwestern University, Evanston, IL, U.S.A.*

[2]*Lewis-Sigler Institute for Integrative Genomics, Princeton University, Princeton, NJ, U.S.A.*

[3]*Ludwig Institute for Cancer Research, Princeton University, Princeton, NJ, U.S.A.*

[4]*Material Measurement Laboratory, National Institute of Standards and Technology, Gaithersburg, MD, U.S.A.*

[5]*Department of Computer Science, Northwestern University, Evanston, IL, U.S.A.*

Graph neural networks (GNNs) have proven effective in understanding and predicting diverse material properties, even when working with limited datasets. An important step in training GNN is to use an appropriate and informative graph embedding that can adequately represent the structural and compositional information in the chemical space. Current graph embeddings consist of composition and structure-agnostic element-level encodings, which are static in nature. This makes it challenging to differentiate between different compounds on the element level, especially for datasets with limited data size, thereby relying more on the complex input and architecture for model training. Here, we present a novel framework

for GNN-based prediction tasks that use dynamic embedding to significantly improve the models' predictive ability on materials properties with limited data size. We evaluated the proposed framework on multiple materials datasets across various domains to find that the model trained using dynamic embedding outperforms the models trained using conventional static embedding and features obtained using a pre-trained model. The proposed framework holds significant potential for expediting artificial intelligence (AI)-driven materials discovery.

Introduction

The field of materials science has experienced an increasing utilization of artificial intelligence (AI) and machine learning (ML) methods, boosting the development of various data-driven models for forward [1–3] and inverse design problems. [4–6] Among these methodologies, graph neural networks (GNNs), in particular, have become increasingly sought-after owing to their ability to aid the discovery process of novel materials and molecules for various applications [7–10]. In recent years, the development of fast and efficient GNN models for forward predictive modeling of materials properties with desired properties has arguably received the most interest for its potential to accelerate materials design [11,12]. Broadly speaking, the quality of the predictive model trained using GNN depends on several factors, including the capability of the model architecture to effectively extract relevant information from the input data, the amount of data available for training the model, and the meaningful information contained within the graph embedding used to represent the chemical compound. There has been ongoing research to improve the model's performance by taking one or more of these

factors into consideration [13–15].

Various works have attempted to improve the performance of the GNN model by modifying the architectures used for predicting material properties such as Crystal Graph Convolutional Neural Networks (CGCNN) [16], Representation Learning from Stoichiometry (Roost) [17], SchNet [18], MatErials Graph Network (MEGNet) [19], DimeNet++ [20], and Atomistic Line Graph Neural Network (ALIGNN) [21]. Several works have also attempted to improve the performance of the model when dealing with limited datasets[22], either via materials property specific feature engineering performed before training the model [23–27] or utilizing the knowledge learned from a trained model on a large dataset to boost the performance of the small dataset using advanced data mining techniques such as feature extraction [28–30] and architectural optimization [31–33]. Additionally, graph-based descriptors and topological index methods have been widely developed for molecular systems, including benzenoid hydrocarbons and molecular trees, with well-known indices such as the Zagreb and Szeged indices. However, these approaches are primarily designed for specific classes of organic molecules characterized by well-defined molecular graphs. Consequently, there is less visibility of works that focus on developing informative input descriptors in the form of graph embeddings for the GNN designed for solids with well-defined structures. Conventionally, a crystalline material, when represented as a graph, comprises of nodes corresponding to constituent atom features and edges corresponding to bond features. As the fundamental node and edge information available to use are limited [34], recent works have tried to improve the model performance by incorporating more information (such as angle-based information [21,35])

or implementing complex components in the model architecture. Moreover, publicly used and available graph embeddings often use atom-level encodings that are composition and structure agnostic. This means that the embeddings represent individual atoms (such as Co or O) the same way, regardless of the molecule's overall structure or complexity [36,37]. For example, the atom-level embedding for cobalt (Co) in a simple molecule like Co_3O_4 (cobalt oxide) would be identical to that in a more complex structure like $LiCoO_2$ (lithium cobalt oxide). This uniform representation does not take into account how the atom's bonding environment or the molecule's composition might affect its chemical properties, which can limit the expressiveness of the embeddings. Additionally, these uniform representations can be problematic, particularly when working with properties that depend heavily on the precise chemical environment, such as electronic, mechanical, or thermal properties. This issue becomes more pronounced when large, diverse datasets are unavailable for model training, as the models have fewer opportunities to effectively extract relevant information. Thus, the problem is not just the structure-agnostic nature of the embeddings, but also the challenge of training effective models in low-data regimes, where simple model training frameworks may not be able to compensate for these limitations.

In this work, we present a novel framework that uses composition and structure aware element-level encodings to represent a chemical compound in a graph neural network (GNN) to improve the predictive performance of the model for materials properties with limited data size. The workflow comparison of the traditional and proposed approach for training graph neural networks is shown in Figure 1. Here, we first apply a GNN architecture that

Figure 1: Comparison of traditional static embedding and proposed dynamic embedding approaches for materials property prediction. (a) Traditional approach: Pre-defined static embeddings are retrieved from pre-calculated look up table. Each element (e.g., Co, Li, O) has identical embedding vectors regardless of its structural environment, resulting in the same representations despite their different crystal structures. (b) Proposed approach: Dynamic embeddings are extracted from a pre-trained structure-aware ALIGNN model that processes atomic positions, bond lengths, and bond angles. Each element receives different embedding vectors depending on its local structural environment, enabling the same element (e.g., Co) to have distinct representations in different materials (Co3O4 vs. LiCoO2).

incorporates composition and structure information as model input to capture the underlying chemistry associated with the existing large data containing crystal structures. The resulting model is then used as an embedding extractor to extract dynamic graph embeddings to represent each input of the target dataset. We use an ALIGNN [21] model as the embedding extractor, as it takes atom, bond, and angle-based information as the model input and has been shown to learn meaningful information from large materials datasets [38]. The ALIGNN-based embedding extractor consists of one embedding layer, followed by four ALIGNN layers and four GCN layers, all connected in a linear sequence. This structure results in a total of nine embedding layers. While the ALIGNN [21] work focuses on learning the chemistry between elements in compounds using conventional CGCNN embeddings (static), our method leverages this learned chemistry from ALIGNN to extract more meaningful representations (dynamic). These representations are expected to better capture interactions and be more effective for downstream tasks. For training the target model, we use Roost [17] as it takes element-level graph embedding as the model input and has been shown to outperform various deep learning (DL) models for training materials properties with limited data points. We compare models trained using the proposed framework with models trained using conventional static embedding and methodologies that use embedding and/or features obtained using a pre-trained model. As, the performance difference between the models trained on the GNN and the conventional DL architectures are already explored in [38], here we mainly compare the results with models trained on the conventional static embedding and methodologies that use embedding and/or features obtained using a pre-trained model.

We observe that the proposed framework can readily accommodate the continuously expanding datasets and evolving model training techniques, facilitating further enhancements to the models. These improvements are anticipated to aid materials science researchers in effectively leveraging data mining techniques [39–41]. This can enable more reliable and accurate screening and identification of potential material candidates, thereby accelerating the materials discovery process.

Results

Datasets We use two datasets of density functional theory (DFT)-computed properties in this work: Materials Project (MP) [42] and Joint Automated Repository for Various Integrated Simulations (JARVIS) [43,44]. MP dataset was downloaded from [21] and the JARVIS dataset from the following figshare link `https://figshare.com/collections/ALIGNN_data/5429274`.

The MP dataset with Formation Energy [21] is used to train embedding extractor model, which is then used to extract dynamic embedding and train small target datasets, which correspond to materials properties with limited data points, to improve their predictive performance. Formation energy is selected as the materials property as it has shown to learn to meaningful representations from large source datasets [28,38]. Materials properties in the JARVIS datasets are used to train target model. Once trained, the target model is used to make predictions and evaluate material properties for compounds in holdout test set. We use dataset without structural polymorphs for a few of the analyses by keeping the most stable

structure available in the database, i.e., data entry corresponding to the lowest formation energy among all compounds with the same composition. The target datasets are randomly split into training, validation, and holdout test sets in the ratio of 80:10:10 with a fixed random seed. The data size corresponding to each materials property in the two datasets are shown in Supplementary Table 1, modifications applied to certain materials properties within the target dataset to align with the model input are shown in Supplementary Table 2, and data distribution for training and test sets for all properties of JARVIS dataset are shown in Supplementary Figure 1 (a) to (ar). We use mean absolute error (MAE) as the primary evaluation metric for all models. Due to the extensive range of materials properties examined in this study and the constraints posed by limited computational resources, we did not explore the aleatoric uncertainty resulting from the random initialization of the models. Additionally, since the performance differences between models trained directly on pre-trained GNNs and those trained via transfer learning—using either fine-tuning or extracted features—have already been explored in prior work [38,45,46], our focus is primarily on comparing downstream models. Specifically, we evaluate models trained using conventional static embeddings against those utilizing embeddings or features derived from pre-trained models.

Dynamic Embedding Extractor We use a structure-aware GNN-based architecture, ALIGNN [21] as the base architecture to train the embedding extractor and extract dynamic embedding as it takes atom, bond, and angle based information as the model input and has been shown to learn meaningful information from the large materials datasets using crystal

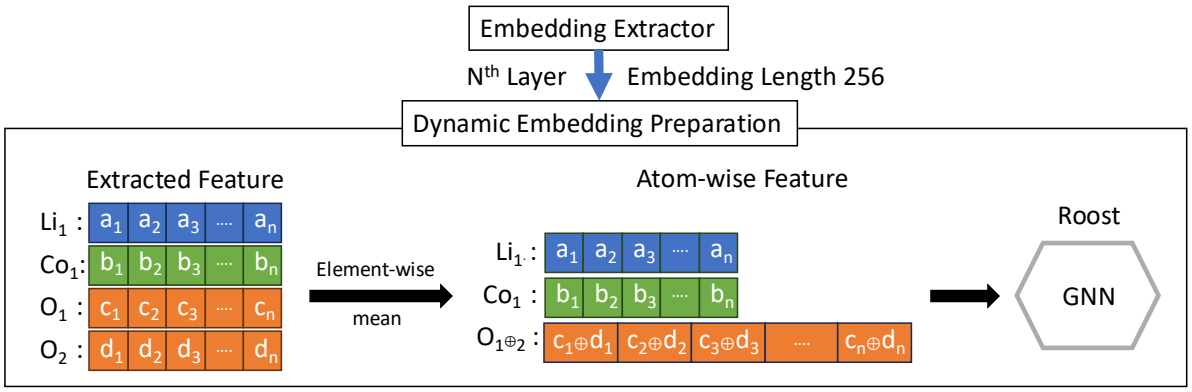

Figure 2: Outline of the operation performed to create dynamic embedding from ALIGNN-based embedding extractor. The embedding comprises of 256 vector dynamic representation of elements. We take element-wise mean (depicted with $\oplus$) of embeddings for the compound with multiple atoms for the same element. The final set of embeddings are used as an input to the graph neural network.

structure information as the model input [38]. For the initial set of input features used to train ALIGNN, please refer to the publication [21]. To extract embeddings from ALIGNN, we design a novel ALIGNN-based Embedding Extractor, shown in Figure 2.

The structure file containing information on the atomic positions and lattice geometry of a chemical compound is divided into atom, bond, and angle based encodings before being fed into an ALIGNN-based embedding extractor, where we perform embedding extraction. The ALIGNN-based embedding extractor consists of one embedding layer, followed by four ALIGNN layers and four GCN layers, all connected in a linear sequence. This structure results in a total of nine embedding layers. We extract the element-level encoding before and after the ALIGNN and GCN (Graph Convolution Network) layers in the ALIGNN ar-

chitecture. We call these encodings dynamic embeddings as they vary depending on the layer they are extracted from and for each element in every compound. After performing embedding extraction, we obtain nine sets of dynamic embeddings, each with a unique 256-vector atom-level representation of the elements of the chemical compound. Due to the nature of the model architecture, all the embeddings extracted from the embedding extractor contain atom, bond, and angle based structure information. For a detailed explanation of the pre-processing of the structure-based encoding associated with the embedding extractor, please refer to the methods section. Note that the concept of dynamic embedding representations—where features extracted from a neural network are used for downstream tasks—has been explored in recent years, existing methods predominantly employ frameworks that extract one-dimensional embeddings which can only be used with feed-forward neural networks [28,47] to train models for the downstream tasks. Our work distinguishes itself by introducing a framework that extracts atom-wise dynamic embedding representations, which are then utilized as inputs for a GNN which requires a more complex form of input for model training. To the best of our knowledge, no previous work has employed this approach, making our framework novel in its application of atom-level embeddings to GNN architectures for enhanced structural representation and improved predictive performance on the downstream task. We show the difference between the traditional framework and proposed framework for feature extraction in Figure 3.

Next, we perform model training using the above-defined set of embeddings as input for the deep graph neural network where we use Roost [17] as the base architecture as it takes

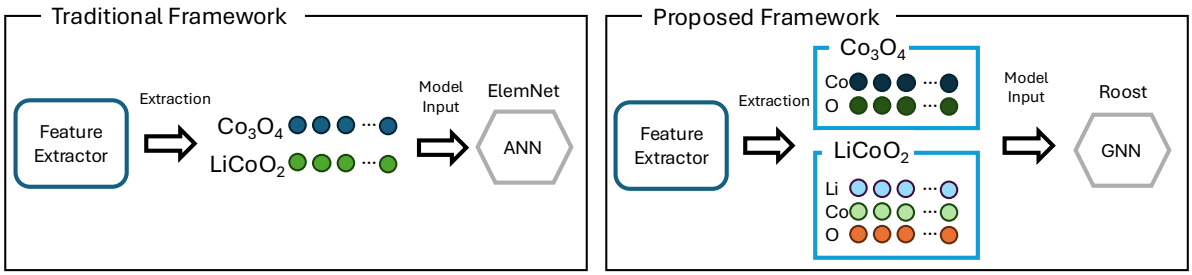

Figure 3: Workflow comparison of the traditional and proposed framework for feature extraction. Traditional methods predominantly employ frameworks with one-dimensional embeddings as the final output which can only be used with artificial neural networks (ANN) to train models for the downstream tasks. Our proposed framework extracts atom-wise dynamic embedding representations, which are then utilized as inputs for a GNN which requires a more complex form of input for model training.

element-level graph embedding as input for model training. We use the formation energy of the JARVIS dataset as the materials property for the property prediction task.

Table 1 shows that, in general, dynamic embeddings obtained from the last layer of the GNN architecture perform better than other layers. Hence, for the rest of the analysis, we only use the dynamic embedding extracted from the last layer to perform model training on the target datasets.

Static Embedding vs. Dynamic Embedding Here, we compare the performance of the model trained on dynamic embeddings against conventional static embeddings using multiple target properties in the JARVIS dataset. For static embeddings, we use OneHot, Magppie [48], CGCNN [16], MEGNet [19], Mat2Vec [49] and Matscholar [50] representations to train the target

Table 1: The table shows the test mean absolute error (MAE) when run on embeddings extracted from different layers of embedding extractor on formation energy of JARVIS dataset.

Layer Number	Validation MAE (eV/atom)	Test MAE (eV/atom)
1	0.107	0.108
2	0.068	0.068
3	0.061	0.059
4	0.057	0.055
5	0.051	0.049
6	0.047	0.046
7	0.046	0.045
8	0.047	0.045
9	0.045	0.043

model which are shown to be meaningful high-dimensional representations of chemical elements [34]. OneHot and Magpie [48] embeddings only take composition-based information into consideration. CGCNN [16] and MEGNet [19] embeddings are derived from crystal graph convolutional neural networks trained to predict materials properties using structure information contained in crystalline materials as input. Mat2Vec [49] and Matscholar [50] are literature word embeddings obtained by using over 3.27 million materials science related abstracts. While OneHot and Magpie utilize domain-driven feature engineering, CGCNN, MEGNet, Mat2Vec, and Matscholar use atom-level embeddings obtained from pre-trained models, representing a form of transfer learning. Note that all models use Roost as the base architecture for training with the respective embeddings (i.e., they correspond to OneHot+Roost, CGCNN+Roost, etc). Interested readers can refer to their respective publications for more details. Table 2 presents the prediction accuracy for 44 target materials properties trained on different sets of embeddings.

Table 2 indicates that models trained using dynamic embedding significantly outper-

Table 2: The table shows the test MAE, and % error change between proposed and best performing static embedding for each of the target materials properties trained on different sets of embeddings using Roost for the prediction task of "Static Embedding v/s Dynamic Embedding".

Property	**Data Size**	**Onehot**	**Magpie** [39]	**CGCNN** [16]	**MEGNET** [19]	**Mat2Vec** [40]	**Matscholar** [41]	**Dynamic (proposed)**	**% Error Change**
E_g OPT (eV)	39760	0.227	0.223	0.239	0.224	0.229	0.219	**0.172**	-21.5
E_f (eV/atom)	39760	0.090	0.086	0.092	0.093	0.087	0.086	**0.043**	-50.0
KLU (Å)	39578	12.19	11.86	12.25	12.24	12.08	11.65	**10.47**	-10.1
Ehull (eV/atom)	39487	0.051	0.052	0.062	0.054	**0.051**	0.052	0.097	90.2
Encut (eV)	39461	148.6	144.9	148.4	142.4	146.0	140.4	**136.5**	-2.8
Magoszi (μ_B)	37574	0.446	0.438	0.470	0.438	0.442	0.429	**0.342**	-20.3
Magout (μ_B)	37932	0.548	0.565	0.574	0.547	0.545	0.531	**0.452**	-14.9
Epsx	31227	29.34	28.68	29.28	28.94	28.72	27.60	**22.85**	-17.2
Epsy	31227	29.09	28.66	28.77	28.87	29.08	27.65	**23.03**	-16.7
Epsz	31227	28.10	27.75	27.87	28.47	28.18	27.02	**22.26**	-17.6
PPF ($\mu W m^{-1} K^{-2}$)	16443	568.8	525.0	545.5	527.6	553.9	521.9	**470.2**	-9.9
NPF ($\mu W m^{-1} K^{-2}$)	16443	570.9	552.8	543.1	543.4	558.4	534.8	**496.8**	-7.1
NSB ($\mu V/K$)	16387	57.31	57.28	56.75	54.92	55.49	53.91	**42.53**	-21.1
PSB ($\mu V/K$)	16400	64.81	63.04	63.38	60.88	64.21	59.04	**49.01**	-17.0
Nem300k (m_0)	15252	0.400	0.417	0.407	0.412	0.422	0.408	**0.371**	-7.3
Pem300k (m_0)	14364	0.686	0.656	0.645	0.624	0.671	0.617	**0.475**	-23.0
ETC33 (GPa)	13195	45.07	45.22	43.51	44.92	43.60	43.44	**34.22**	-21.2
ETC22 (GPa)	13098	43.59	41.78	42.71	40.95	40.86	41.34	**32.79**	-19.8
ETC11 (GPa)	13107	44.59	43.26	42.79	42.69	42.58	41.59	**34.23**	-17.7
BulkKV (GPa)	12789	18.60	18.21	18.46	19.22	17.46	17.31	**13.74**	-20.6
ETC13 (GPa)	13056	18.55	18.16	17.84	17.80	18.08	17.30	**15.10**	-12.7
ETC12 (GPa)	12895	22.11	20.85	21.91	21.61	20.67	20.62	**17.61**	-14.6
Poisson	12676	0.234	0.236	0.222	0.204	0.223	0.230	**0.187**	-8.3
ShearGV (GPa)	12179	13.75	12.82	12.86	12.87	13.00	12.77	**9.89**	-22.6
E_g MBJ (eV)	12281	0.410	0.379	0.401	0.369	0.371	0.345	**0.285**	-17.4
ETC44 (GPa)	11863	18.98	18.76	18.27	18.14	17.96	17.98	**14.34**	-20.2
AvgME (electron mass unit)	12637	0.100	0.103	0.098	0.111	0.099	0.097	**0.083**	-14.4
AvgMH (electron mass unit)	12637	0.164	0.162	0.170	0.161	0.176	0.159	**0.130**	-18.2
ETC55 (GPa)	11673	16.61	16.50	16.26	15.34	15.46	15.41	**12.40**	-19.2
ETC66 (GPa)	11538	17.14	16.51	16.49	16.36	16.48	16.31	**12.79**	-21.6
Mepsx	11380	32.79	32.26	31.59	30.94	32.00	30.62	**27.36**	-10.6
Mepsy	11380	32.65	30.93	31.25	30.61	31.14	31.50	**26.62**	-13.0
Mepsz	11380	30.59	29.59	30.25	31.30	30.37	28.91	**25.02**	-13.5
MaxM (cm^{-1})	9156	55.51	50.72	53.38	54.17	52.65	55.19	**41.63**	-17.9
MinM (cm^{-1})	8644	21.55	20.03	20.68	20.34	19.87	20.56	**14.95**	-24.8
MaxEFG (10^{21} Vm^{-2})	7286	32.33	30.58	28.83	30.08	30.23	29.52	**23.27**	-19.3
Spillage	8755	0.415	0.398	0.407	0.394	0.402	0.406	**0.343**	-12.9
SLME (%)	5877	6.479	5.900	5.764	6.289	5.867	5.700	**4.948**	-13.2
PMEij (cm^{-2})	2711	0.134	0.116	0.133	0.118	0.129	0.127	**0.112**	-3.4
PMDi	2649	20.63	19.05	18.60	18.63	18.83	19.41	**16.07**	-13.6
PMDiEl (ε_{11})	2592	2.959	3.167	3.410	3.184	2.866	2.948	**2.631**	-8.2
PMDiIo (ε_{11})	2496	3.864	3.516	3.399	3.841	3.434	3.465	**3.037**	-10.7
PMDij (cm^{-1})	1885	8.878	8.702	7.900	8.461	7.954	7.763	**5.771**	-25.7
Exfoli (meV/atom)	499	48.83	42.86	47.26	42.83	43.37	40.78	**40.62**	-0.4

form the models trained using conventional static embeddings in 43/44 cases, i.e., in ≈ 98 % of the cases. Although we observe improvement across all the target materials properties, the most improvement (i.e., ≈ 50 %) is shown with the formation energy as the target property (0.086 to 0.043 eV/atom). The improvement can be attributed to the fact that the embedding extractor was trained using the formation energy as the materials property. The proposed model performed worse only for Ehull as the materials property. Ehull is unique among our tested properties as it represents relative stability rather than an intrinsic material property. It measures the energy difference between a compound and the thermodynamically stable phase-separated state. This relative nature may explain why embeddings trained on absolute formation energy do not transfer as effectively. It would be interesting to explore why the proposed framework performed worse only for Ehull by analyzing its relation with the embeddings in future work by training embedding extractors directly on convex-hull–related objectives or by using multi-task learning frameworks that jointly optimize for both absolute and relative thermodynamic properties. Overall, the results illustrate that the embedding extractor can learn and extract useful and widely applicable embeddings during model training on source data and improve the performance of the target model trained on materials properties with limited data size.

Comparison Against Other Methods In this section, we investigate the performance of the proposed framework against other well-known deep neural networks and methodologies that use embedding and/or features obtained using a pre-trained model to improve the performance of the model for small datasets, i.e., CrabNet [51], and AtomSets [45]. CrabNet [51]

uses literature word embedding mat2vec [49] as input representation with attention-based network to train the model. AtomSets [45] comprises compositional and structural descriptors extracted from pre-trained MEGNet [19] model. These features serve as input to train conventional DL models, thereby implementing transfer learning similar to the approach proposed in this work. We also compare the proposed framework with conventional deep learning (ElemNet [47]) and traditional machine learning (AutoML [52]) models. As they require one-dimensional vector representation as input, we use elemental fraction (EF) [47], physical attributes (PA) [48], and atom-wise averaged value of dynamic embedding as the inputs to the model. As we have multiple trained models for AtomSets, ElemNet, and AutoML, we use the model with the least test MAE to compare the performance in Table 3 and show the rest of the results in Supplementary Tables 3, and 4.

Table 3 indicates that models trained using dynamic embedding outperform other embedding and/or features based methods in 33/44 cases, i.e., in $\approx$ 75 % of the cases. For the remaining 11 properties, the model trained on ElemNet using atom-wise averaged value of dynamic embedding performed the best for 7/11 cases. It is interesting that the methods that incorporate dynamic embedding in one form or another can outperform other methods in 40/44 cases, i.e., in $\approx$ 91 % of the cases. We also observe close error values for some of the materials properties between the proposed method and ElemNet trained using atom-wise averaged value of dynamic embedding. However, as both the models are trained using default hyperparameter settings, the proposed methods train for 250 epochs, and ElemNet trains with patience of 100 epochs (i.e., stop the model training if the validation loss does not

Table 3: The table shows the test MAE, and % error change between proposed method and (1) best performing existing method and (2) method using embedding extracted from ALIGNN as input for each of the target materials properties trained on different models for the prediction task of "Comparison Against Other Methods".

Property	Data Size	Existing		Embedding			% Error Change (Existing)	% Error Change (Embedding)
		CrabNet [42]	AtomSets [36]	AutoML [43]	ElemNet [38]	Dynamic (proposed)		
E_g OPT (eV)	39760	0.302	0.237	0.283	**0.171**	0.172	-27.4	0.6
E_f (eV/atom)	39760	0.108	0.066	0.075	0.043	**0.042**	-36.4	-2.3
KLU (Å)	39578	10.50	12.10	10.62	10.23	10.47	-0.3	2.3
Ehull (eV/atom)	39487	**0.089**	0.168	0.142	0.156	0.097	9.0	-31.7
Encut (eV)	39461	**124.0**	164.7	152.9	138.1	136.5	10.1	-1.2
Magoszi (μ_B)	37574	0.504	0.431	0.537	0.373	**0.342**	-20.6	-8.3
Magout (μ_B)	37932	0.641	0.614	0.665	0.466	**0.452**	-26.4	-3.0
Epsx	31227	25.10	31.32	25.98	23.96	**22.85**	-9.0	-4.6
Epsy	31227	25.40	31.69	26.90	23.61	**23.03**	-9.3	-2.5
Epsz	31227	24.60	31.74	25.84	22.98	**22.26**	-9.5	-3.1
PPF ($\mu W m^{-1} K^{-2}$)	16443	487.0	575.1	526.9	471.4	**470.2**	-3.4	-0.3
NPF ($\mu W m^{-1} K^{-2}$)	16443	520.0	601.9	541.4	**484.7**	496.8	-4.5	2.5
NSB ($\mu V/K$)	16387	50.50	57.70	50.28	45.67	**42.53**	-15.8	-6.9
PSB ($\mu V/K$)	16400	51.50	62.27	56.78	49.69	**49.01**	-4.8	-1.4
Nem300k (m_0)	15252	0.389	0.475	0.438	**0.357**	0.371	-4.6	3.9
Pem300k (m_0)	14364	0.658	0.642	0.633	0.501	**0.475**	-26.0	-5.2
ETC33 (GPa)	13195	38.60	47.62	40.82	35.74	**34.22**	-11.3	-4.3
ETC22 (GPa)	13098	36.40	44.58	40.05	34.79	**32.79**	-9.9	-5.7
ETC11 (GPa)	13107	37.70	45.67	40.67	35.07	**34.23**	-9.2	-2.4
BulkKV (GPa)	12789	14.50	19.50	17.83	14.15	**13.74**	-5.2	-2.9
ETC13 (GPa)	13056	15.30	20.41	16.87	15.63	**15.10**	-1.3	-3.4
ETC12 (GPa)	12895	18.80	23.53	19.72	18.02	**17.61**	-6.3	-2.3
Poisson	12676	**0.168**	0.234	0.181	0.173	0.187	11.3	8.1
ShearGV (GPa)	12179	12.30	14.60	12.39	10.27	**9.89**	-19.6	-3.7
E_g MBJ (eV)	12281	0.489	0.428	0.406	0.354	**0.285**	-33.4	-19.5
ETC44 (GPa)	11863	15.60	19.56	16.47	14.58	**14.34**	-8.1	-1.6
AvgME (electron mass unit)	12637	0.103	0.111	0.113	0.090	**0.083**	-19.4	-7.8
AvgMH (electron mass unit)	12637	0.171	0.178	0.183	0.142	**0.130**	-24.0	-8.5
ETC55 (GPa)	11673	14.20	16.61	15.25	12.50	**12.40**	-12.7	-0.8
ETC66 (GPa)	11538	14.60	17.20	14.61	13.21	**12.79**	-12.4	-3.2
Mepsx	11380	28.50	36.11	30.58	27.61	**27.36**	-4.0	-0.9
Mepsy	11380	27.40	35.07	29.69	26.68	**26.62**	-2.8	-0.2
Mepsz	11380	27.90	34.92	29.28	25.88	**25.02**	-10.3	-3.3
MaxM (cm^{-1})	9156	60.60	62.70	55.58	57.47	**41.63**	-31.3	-25.1
MinM (cm^{-1})	8644	20.60	24.92	23.12	15.70	**14.95**	-27.4	-4.8
MaxEFG (10^{21} Vm^{-2})	7286	27.80	31.70	27.65	25.82	**23.27**	-16.3	-9.9
Spillage	8755	0.372	0.451	0.375	0.350	**0.343**	-7.8	-2.0
SLME (%)	5877	6.460	7.058	6.184	5.690	**4.948**	-23.4	-13.0
PMEij (cm^{-2})	2711	0.112	0.132	0.133	**0.101**	0.112	0.0	10.9
PMDi	2649	16.50	19.73	19.53	**15.07**	16.07	-2.6	6.6
PMDiEl (ε_{11})	2592	2.800	3.088	3.248	2.733	**2.631**	-6.0	-3.7
PMDiIo (ε_{11})	2496	3.480	3.935	3.568	3.330	**3.037**	-12.7	-8.8
PMDij (cm^{-1})	1885	6.560	9.407	8.305	**5.685**	5.771	-12.0	1.5
Exfoli (meV/atom)	499	40.10	**30.37**	36.92	42.07	40.62	33.8	10.0

improve after 100 epochs and save the model with the best validation error). This can cause ElemNet to train for more epochs, leading to longer training time. It would be interesting to see if the performance of the proposed method improves if we increase the number of epochs to train the model. Additionally, as the AtomSets framework provides different features for composition and structure based information, analyzing the results provided in Supplementary Table 4 can provide insights on what information is beneficial to train which materials properties. Overall, the results demonstrate that the proposed framework performs better with respect to other embeddings and other methodologies that use embedding and/or features obtained using a pre-trained model to improve the performance of the model for small datasets.

Including Structural Polymorphs in Dataset In the previous sections, we used a dataset without structural polymorphs entries. This is because the model used to train the small target dataset was designed to work with static, structure-agnostic embeddings. These embeddings cannot differentiate between various structural polymorphs of the same composition, so polymorph entries had to be removed during model training. However, as the proposed dynamic embedding framework uses different sets of element-level encodings for each element in every compound, by using them as embeddings for training the model, we can effectively convert the composition-based models into structure-based models and thereby mitigate the shortcomings of models trained on composition-based inputs only. We compare the performance of the proposed framework with AtomSets, which uses structure-based features obtained using a pre-trained model along with ElemNet and AutoML trained

using atom-wise averaged value of dynamic embedding. We could not use structure-based representations such as classical force-field inspired descriptors (CFID) [53] and voronoi tessellations based descriptors [54] as model input to ElemNet and AutoML as their respective libraries failed to featurize substantial amount of training, validation and test sets. Moreover, using only structure information as input to the deep learning and traditional machine learning model has been shown not to improve the performance of the model [55]. As we have multiple trained models for AtomSets, we use the model with the least test MAE to compare the performance in Table 4 and show the rest of the results in Supplementary Tables 5.

Table 4 indicates that models trained using dynamic embedding outperform other embedding and/or features based methods in 34/44 cases, i.e., in ≈ 77 % of the cases. For the remaining 10 properties, the model trained on ElemNet using atom-wise averaged value of dynamic embedding performed the best for 10/10 cases. Similar to the observations made in the previous section, the methods that incorporate dynamic embedding in one form or another can outperform other methods, and for the materials properties with close error values between the proposed method and ElemNet trained using atom-wise averaged value of dynamic embedding, we can explore the effect of epochs for training the model with the performance. We also observe a clear benefit of using dynamic embeddings as features when working with datasets that include structural polymorphs. These embeddings capture more meaningful structural differences between compounds with identical compositions but distinct crystal structures, as evidenced by improved model performance. It is quite encouraging to observe that the model that was originally designed to take composition-based inputs only

Table 4: The table shows the test MAE, and % error change between proposed method and (1) best performing existing method and (2) method using embedding extracted from ALIGNN as input for each of the target materials properties trained on different models for the prediction task of "Including Structural Polymorphs in Dataset".

		Existing	Embedding			% Error	% Error
	Data	AtomSets	AutoML	ElemNet	Dynamic	Change	Change
Property	Size	[36]	[43]	[38]	(proposed)	(Existing)	(Embedding)
E_g OPT (eV)	54934	0.227	0.265	0.158	**0.140**	-38.3	-11.4
E_f (eV/atom)	54934	0.075	0.073	0.047	**0.046**	-38.7	-2.1
KLU (Å)	54606	11.45	9.99	**9.41**	9.92	-4.1	-13.4
Ehull (eV/atom)	54546	0.191	0.231	0.159	**0.075**	-60.7	-52.8
Encut (eV)	54469	161.0	152.4	129.9	**122.0**	-24.2	-6.1
Magoszi (μ_B)	51443	0.431	0.504	0.297	**0.289**	-32.9	-2.7
Magout (μ_B)	52125	0.588	0.700	0.446	**0.430**	-26.9	-3.6
Epsx	43760	30.07	24.97	21.81	**21.48**	-28.6	-1.5
Epsy	43760	29.32	25.00	22.10	**21.31**	-27.3	-3.6
Epsz	43760	28.78	24.76	21.94	**21.93**	-23.8	0.0
PPF ($\mu W m^{-1} K^{-2}$)	22692	585.5	490.0	**445.3**	457.2	-21.9	2.7
NPF ($\mu W m^{-1} K^{-2}$)	22692	587.1	518.8	**468.2**	481.9	-17.9	2.9
NSB ($\mu V/K$)	22609	58.56	51.43	43.69	**42.70**	-27.1	-2.3
PSB ($\mu V/K$)	22634	59.62	55.05	45.84	**45.35**	-23.9	-1.1
Nem300k (m_0)	20899	0.439	0.460	0.318	**0.310**	-29.4	-2.5
Pem300k (m_0)	19730	0.630	0.660	**0.454**	0.497	-21.1	9.5
ETC33 (GPa)	19530	48.78	38.44	34.45	**34.44**	-29.4	0.0
ETC22 (GPa)	19416	44.83	38.21	34.35	**32.73**	-27.0	-4.7
ETC11 (GPa)	19381	47.58	38.18	35.03	**33.89**	-28.8	-3.3
BulkKV (GPa)	19003	20.46	19.14	14.30	**12.90**	-37.0	-9.8
ETC13 (GPa)	19023	22.06	18.73	15.38	**14.90**	-32.5	-3.1
ETC12 (GPa)	18945	25.21	20.92	18.61	**18.12**	-28.1	-2.6
Poisson	18902	0.218	0.161	**0.143**	0.160	-26.6	11.9
ShearGV (GPa)	18004	12.607	10.787	8.878	**8.683**	-31.1	-2.2
E_g MBJ (eV)	17578	0.416	0.445	0.273	**0.243**	-41.6	-11.0
ETC44 (GPa)	17231	19.62	15.83	14.40	**14.03**	-28.5	-2.6
AvgME (electron mass unit)	17181	0.106	0.136	0.081	**0.080**	-24.5	-1.2
AvgMH (electron mass unit)	17181	0.157	0.192	0.121	**0.115**	-26.8	-5.0
ETC55 (GPa)	16859	16.34	14.36	**12.17**	12.27	-24.9	0.8
ETC66 (GPa)	16650	15.76	13.63	11.88	**11.71**	-25.7	-1.4
Mepsx	16349	31.23	28.36	24.69	**23.36**	-25.2	-5.4
Mepsy	16349	34.46	28.72	23.91	**23.24**	-32.6	-2.8
Mepsz	16349	34.09	29.36	**24.02**	24.55	-28.0	2.2
MaxM (cm^{-1})	13379	52.58	81.74	46.05	**35.12**	-33.2	-23.7
MinM (cm^{-1})	12678	30.83	30.28	20.59	**20.45**	-33.7	-0.7
MaxEFG (10^{21} Vm^{-2})	11506	31.51	26.47	22.66	**22.05**	-30.0	-2.7
Spillage	11203	0.464	0.408	**0.349**	0.351	-24.4	0.6
SLME (%)	8923	6.228	5.849	4.996	**4.445**	-28.6	-11.0
PMEij (cm^{-2})	4745	0.143	0.139	0.109	**0.098**	-31.5	-10.1
PMDi	4600	23.57	22.61	15.35	**15.27**	-35.2	-0.5
PMDiEl (ε_{11})	4495	3.798	4.035	**3.173**	3.585	-5.6	13.0
PMDiIo (ε_{11})	4307	4.249	3.591	3.160	**3.122**	-26.5	-1.2
PMDij (cm^{-1})	3290	14.29	13.87	10.05	**9.92**	-30.6	-1.3
Exfoli (meV/atom)	801	44.14	44.66	**38.83**	45.42	2.9	17.0

as input can outperform other methods when including structure-based information. Additionally, we performed model training by incorporating an additional set of meaningful information in the dynamic embedding to train Roost to observe if the current set of embedding works well with other chemical information associated with a compound. The dynamic embedding uses different element-level encoding to represent each element of every compound. However, as we know that the ALIGNN uses bond and angle based features to train the model, it can also be extracted from a given layer in the architecture. As each compound has a different number of bond and angle information associated with it, we take the average across all that information and create one-dimensional compound-level encoding, which will be different for every compound. Next, we concatenate this compound-level encoding with element-level encoding to create a more informative dynamic embedding. We then use this large dynamic embedding that combines compound-level encoding with element-level encoding as the model input to train the model. Additionally, we can also concatenate static embeddings with dynamic embeddings to improve the results. Hence, we try combinations for a couple of static and dynamic embeddings with element-level encoding and compound-level encoding for the analysis. The prediction accuracy of the proposed framework with different combinations of embeddings as shown in Supplementary Table 6 indicates that adding compound-level encodings, i.e., bond and angle representation, to the element-level encodings helps improve the model's performance. On the other hand, training model using the combination of dynamic embeddings with static embedding does not contribute towards improving the performance of the model. This, along with the other results, demonstrates

that the proposed framework can significantly and consistently help improve the prediction of the materials properties across various domains with limited datasets, thereby potentially saving time and resources in the process of future materials discovery.

Discussion

In this paper, we presented a novel framework that uses composition and structure aware element-level encodings to represent a compound in a graph neural network (GNN) to build accurate models with limited datasets for improved materials property prediction across a wide variety of materials properties. Additionally, our framework offers a principled method for incorporating structural polymorphs during training, even in neural network architectures originally designed to accept only a single structure per composition. To show the benefit of the proposed approach, we built an embedding extractor using a GNN architecture ALIGNN that incorporates composition and structure based information as input and formation energy of the MP dataset as the materials property. This model was used to extract dynamic embeddings to train a target model on 44 different materials properties to find that the proposed framework produces accurate and robust models for datasets with limited data size. We compare the performance of the target model trained using dynamic embedding with conventional static embedding and methodologies that use embedding and/or features obtained using a pre-trained model. We also observed that adding compound-level encodings, i.e., bond and angle representation, to the element-level encodings helps improve the model's performance. On the other hand, training model using the combination of dynamic embed-

dings with static embedding does not contribute towards improving the performance of the model. Hence, one can also explore different sets of embeddings to train the target dataset or use more sophisticated architectures for the target model in a bid to boost the performance of the target model for a specific materials property. To check the robustness of the proposed framework even further, we perform additional experiments using the formation energy of the JARVIS dataset to examine the other applications of the dynamic embedding obtained using the embedding extractor. First, we performed model training using other deep neural network architecture that uses embeddings as input, i.e., CrabNet [51] to see if dynamic embedding produces similar performance improvements as observed in the Roost. The model trained on original mat2vec embeddings gives a test MAE of 0.108 eV/atom, whereas the model trained on proposed dynamic embeddings gives a test MAE of 0.066 eV/atom, which is ≈ 39 % improvement in accuracy. This shows that dynamic embedding can be applied to other types of deep neural network architectures that take embeddings as input for datasets with limited data size. Next, we evaluate the computational overhead associated with the proposed dynamic embeddings relative to conventional static embeddings (matscholar) when training the Roost model, using formation energy (a widely studied materials property)from the JARVIS dataset, using 10-fold cross-validation in Supplementary Table 7. We find that both the per-epoch training time and the total training time for Roost models employing dynamic embeddings are comparable to those using static embeddings, despite the substantial gains in predictive accuracy achieved with dynamic embeddings. The principal additional cost arises during the initial embedding extraction stage; however, this step is performed only

once and the resulting embeddings can be reused across multiple downstream tasks without re-extraction. Consequently, dynamic embeddings provide a computationally efficient and reusable representation that consistently enhances model performance across diverse target material properties. At last, we present the training time per epoch (in seconds) and the total number of epochs for each target material property trained using ElemNet or Roost in Tables 3 and 4 in Supplementary Table 8. Although both Roost and ElemNet perform similarly, with Roost showing slightly better average performance, it is notable that Roost takes more time per epoch, while ElemNet requires more epochs to train.

The proposed framework is thus flexible and robust, and can leverage a wide variety of deep neural networks to improve upon the performance using a rich set of embeddings. This model is primarily designed for solids with well-defined structures. However, existing studies [28,38] have demonstrated that transfer learning can be applied to various materials properties across domains with limited datasets making such approaches agnostic to the type of downstream application. This indicates potential transferability to materials with less precise structures, such as polymer melts or ionic liquids. Previous works [56,57] that have applied TL to materials with less precise structures have used simple ML/DL techniques to train the models. Extending this approach to more recent models, such as the one proposed in our work, could offer significant improvements and represents a promising direction for future research. The proposed framework is expected to easily adapt to ever-increasing data, ever-advancing model training techniques, and other scientific domains beyond materials science. Moreover, the presented framework is conceptually easy to implement, understand,

use, and build upon. For future work, it would be interesting to explore the effect on the performance of the target model when materials properties other than formation energy are used as the material property and GNN architecture other than ALIGNN is used for training the embedding extractor. It would also be interesting to see the effect of dynamic embedding when used as input for the state-of-the-art GNN architecture or when training even larger target datasets. Additionally, we can also explore the uncertainty associated with the materials property prediction by incorporating neural network components that help perform uncertainty estimation, such as dropout within the network architecture, or by creating an ensemble model using multiple graph neural networks. One can also explore different combinations of embeddings to train the neural network or use more sophisticated neural network architectures for the target model in a bid to boost the performance of the target model for a specific materials property.

Methods

Embedding Extractor In this work, we implement an embedding extractor to extract element-level encoding from DFT-relaxed structures to train the target model. The model pre-trained on ALIGNN using formation energy from the MP dataset is used as an embedding extractor to extract atom-based information from a given layer, each containing a variable number of rows depending on the number of the atom information present in the input file and 256 columns as embedding for each row. For example, let us consider a hypothetical compound $W_w X_x Y_y Z_z$ where $w + x + y + z = Q$ and the number of unique elements is 4.

Next, we extract the embeddings before and after every ALIGNN and GCN layer, where the dimensions of the extracted embeddings will be (Q, 256). We then take the element-wise mean of all embeddings across each column to create a (4, 256) embedding representation where each row represents the element-wise embedding for a given compound of the target dataset. The extracted embeddings from a given layer can then be used as a dynamic embedding for any GNN network that uses embedding as a form of model input.

Network Settings and Model Architecture Roost [17] and ALIGNN [21] were implemented using Pytorch. The hyperparameters used in the Roost comprise of the following: Adam as the optimizer with weight decay parameter of 10^{-6}, LeakyReLU as the base activation function, mini-batch size of 128, and learning rate as 0.0003. We train all Roost models for 250 epochs as done in the original work [17]. The hyperparameters used in the ALIGNN comprise of the following: Adaptive Moment Estimation with decoupled weight decay (AdamW) as the optimizer with normalized weight decay of 10^{-5}, Sigmoid Linear Unit (SiLU) as the base activation function, mini-batch size of 64, and learning rate as 0.001. We train all ALIGNN models for 300 epochs with a fixed random seed as done in the original work [21]. Readers interested in in-depth hyperparameter settings for Roost and ALIGNN models are referred to those publications [17,21] for details. We use mean absolute error (MAE) as the loss function as well as the primary evaluation metric for all models. Each model is trained on a single Tesla V100-PCIE-16GB Graphics processing unit (GPU). For all other downstream models we used the default hyperparameters and training procedures for training the models. For all other downstream models, we retained the default hyperparameters and training procedures

recommended in the original publications and reference codebases referenced in the code availability section.

Data availability The datasets used in this paper are publicly available from the corresponding websites- MP [42] from `https://materialsproject.org/` and JARVIS [43,44] from `https://jarvis.nist.gov`.

Code availability The code for extracting dynamic embeddings from the pre-trained ALIGNN model used in the proposed framework is available at `https://github.com/GuptaVishu2002/DynamicEmbedding`. For the downstream prediction models, we used publicly available implementations from the following repositories: Roost (`https://github.com/CompRhys/roost`), ElemNet (`https://github.com/NU-CUCIS/ElemNet`), CrabNet (`https://github.com/anthony-wang/CrabNet`), AtomSets (`https://github.com/materialyzeai/maml`), and AutoML (https://github.com/automl/au sklearn).

References

1. Dane Morgan and Ryan Jacobs. Opportunities and challenges for machine learning in materials science. Annual Review of Materials Research, 50:71–103, 2020.

2. Pascal Friederich, Florian Häse, Jonny Proppe, and Alán Aspuru-Guzik. Machine-learned potentials for next-generation matter simulations. Nature Materials, 20(6):750–761, 2021.

3. Julia Westermayr, Michael Gastegger, Kristof T Schütt, and Reinhard J Maurer. Perspective on integrating machine learning into computational chemistry and materials science. The Journal of Chemical Physics, 154(23):230903, 2021.

4. Yuwei Mao, Mahmudul Hasan, Arindam Paul, Vishu Gupta, Kamal Choudhary, Francesca Tavazza, Wei-keng Liao, Alok Choudhary, Pinar Acar, and Ankit Agrawal. An ai-driven microstructure optimization framework for elastic properties of titanium beyond cubic crystal systems. npj Computational Materials, 9(1):111, 2023.

5. Ghanshyam Pilania. Machine learning in materials science: From explainable predictions to autonomous design. Computational Materials Science, 193:110360, 2021.

6. Rui Jiao, Wenbing Huang, Peijia Lin, Jiaqi Han, Pin Chen, Yutong Lu, Yang Liu. Crystal structure prediction by joint equivariant diffusion Advances in Neural Information Processing Systems, 36:17464–17497, 2023.

7. Rampi Ramprasad, Rohit Batra, Ghanshyam Pilania, Arun Mannodi-Kanakkithodi, and Chiho Kim. Machine learning in materials informatics: recent applications and prospects. npj Computational Materials, 3(1):54, dec 2017.

8. Ankit Agrawal and Alok Choudhary. Deep materials informatics: Applications of deep learning in materials science. MRS Communications, 9(3):779–792, 2019.

9. Vishu Gupta, Wei-keng Liao, Alok Choudhary, and Ankit Agrawal. Evolution of artificial intelligence for application in contemporary materials science. MRS Communications, pages 1–10, 2023.

10. Kamal Choudhary, Daniel Wines, Kangming Li, Kevin F. Garrity, Vishu Gupta, Aldo H. Romero, Jaron T. Krogel, Kayahan Saritas, Addis Fuhr, Panchapakesan Ganesh, Paul R. C. Kent, Keqiang Yan, Yuchao Lin, Shuiwang Ji, Ben Blaiszik, Patrick Reiser, Pascal Friederich, Ankit Agrawal, Pratyush Tiwary, Eric Beyerle, Peter Minch, Trevor David Rhone, Ichiro Takeuchi, Robert B. Wexler, Arun Mannodi-Kanakkithodi, Elif Ertekin, Avanish Mishra, Nithin Mathew, Mitchell Wood, Andrew Dale Rohskopf, Jason Hattrick-Simpers, Shih-Han Wang, Luke E. K. Achenie, Hongliang Xin, Maureen Williams, Adam J. Biacchi, Francesca Tavazza JARVIS-Leaderboard: a large scale benchmark of materials design methods. npj Computational Materials, 10(1):93 , 2023.

11. Kamal Choudhary, Brian DeCost, Chi Chen, Anubhav Jain, Francesca Tavazza, Ryan Cohn, Cheol Woo Park, Alok Choudhary, Ankit Agrawal, Simon J. L. Billinge, Elizabeth Holm, Shyue Ping Ong, Chris Wolverton Recent advances and applications of deep learning methods in materials science. npj Computational Materials, 8(1):59, 2022.

12. Xingyue Shi, Linming Zhou, Yuhui Huang, Yongjun Wu, Zijian Hong A review on the applications of graph neural networks in materials science at the atomic scale Materials Genome Engineering Advances, 2(2):e50, 2024.

13. Zhenyao Fang, Qimin Yan Towards accurate prediction of configurational disorder properties in materials using graph neural networks npj Computational Materials, 10(1):91, 2024.

14. Hongwei Du, Jiamin Wang, Jian Hui, Lanting Zhang, Hong Wang, DenseGNN: universal

and scalable deeper graph neural networks for high-performance property prediction in crystals and molecules npj Computational Materials, 10(1):292, 2024.

15. Reshma Devi, Keith T Butler, Gopalakrishnan Sai Gautam, Optimal pre-train/fine-tune strategies for accurate material property predictions npj Computational Materials, 10(1):300, 2024.

16. Tian Xie and Jeffrey C. Grossman. Crystal graph convolutional neural networks for an accurate and interpretable prediction of material properties. Phys. Rev. Lett., 120:145301, Apr 2018.

17. Rhys EA Goodall and Alpha A Lee. Predicting materials properties without crystal structure: Deep representation learning from stoichiometry. Nature communications, 11(1):1–9, 2020.

18. Kristof Schütt, Pieter-Jan Kindermans, Huziel Enoc Sauceda Felix, Stefan Chmiela, Alexandre Tkatchenko, and Klaus-Robert Müller. Schnet: A continuous-filter convolutional neural network for modeling quantum interactions. Advances in neural information processing systems, 30, 2017.

19. Chi Chen, Weike Ye, Yunxing Zuo, Chen Zheng, and Shyue Ping Ong. Graph networks as a universal machine learning framework for molecules and crystals. Chemistry of Materials, 31(9):3564–3572, 2019.

20. Johannes Klicpera, Shankari Giri, Johannes T Margraf, and Stephan Günnemann. Fast and uncertainty-aware directional message passing for non-equilibrium molecules. arXiv preprint arXiv:2011.14115, 2020.

21. Kamal Choudhary and Brian DeCost. Atomistic line graph neural network for improved materials property predictions. npj Computational Materials, 7(1):1–8, 2021.

22. Dipendra Jha, Vishu Gupta, Wei-keng Liao, Alok Choudhary, and Ankit Agrawal. Moving closer to experimental level materials property prediction using ai. Scientific reports, 12, 2022.

23. Atsuto Seko, Atsushi Togo, Hiroyuki Hayashi, Koji Tsuda, Laurent Chaput, and Isao Tanaka. Prediction of low-thermal-conductivity compounds with first-principles anharmonic lattice-dynamics calculations and bayesian optimization. Physical Review Letters, 115(20):205901, 2015.

24. Luca M Ghiringhelli, Jan Vybiral, Sergey V Levchenko, Claudia Draxl, and Matthias Scheffler. Big data of materials science: Critical role of the descriptor. Physical Review Letters, 114(10):105503, 2015.

25. Joohwi Lee, Atsuto Seko, Kazuki Shitara, Keita Nakayama, and Isao Tanaka. Prediction model of band gap for inorganic compounds by combination of density functional theory calculations and machine learning techniques. Physical Review B, 93(11):115104, 2016.

26. Austin D Sendek, Qian Yang, Ekin D Cubuk, Karel-Alexander N Duerloo, Yi Cui, and Evan J Reed. Holistic computational structure screening of more than 12000 candidates

for solid lithium-ion conductor materials. Energy & Environmental Science, 10(1):306–320, 2017.

27. Yuwei Mao, Shahriyar Keshavarz, Vishu Gupta, Andrew CE Reid, Wei-keng Liao, Alok Choudhary, and Ankit Agrawal. Ai for learning deformation behavior of a material: Predicting stress-strain curves 4000x faster than simulations. In 2023 International Joint Conference on Neural Networks (IJCNN), pages 1–8. IEEE, 2023.

28. Vishu Gupta, Kamal Choudhary, Francesca Tavazza, Carelyn Campbell, Wei-keng Liao, Alok Choudhary, and Ankit Agrawal. Cross-property deep transfer learning framework for enhanced predictive analytics on small materials data. Nature communications, 12(1):1–10, 2021.

29. Vishu Gupta, Kamal Choudhary, Yuwei Mao, Kewei Wang, Francesca Tavazza, Carelyn Campbell, Wei-keng Liao, Alok Choudhary, and Ankit Agrawal. Mppredictor: An artificial intelligence-driven web tool for composition-based material property prediction. Journal of Chemical Information and Modeling, 63(7):1865–1871, 2023.

30. Vishu Gupta, Wei-keng Liao, Alok Choudhary, and Ankit Agrawal. Pre-activation based representation learning to enhance predictive analytics on small materials data. In 2023 International Joint Conference on Neural Networks (IJCNN), pages 1–8. IEEE, 2023.

31. Vishu Gupta, Wei-keng Liao, Alok Choudhary, and Ankit Agrawal. Brnet: Branched residual network for fast and accurate predictive modeling of materials properties. In

Proceedings of the 2022 SIAM International Conference on Data Mining (SDM), pages 343–351. SIAM, 2022.

32. Vishu Gupta, Alec Peltekian, Wei-keng Liao, Alok Choudhary, and Ankit Agrawal. Improving deep learning model performance under parametric constraints for materials informatics applications. Scientific Reports, 13(1):9128, 2023.

33. Vishu Gupta, Youjia Li, Alec Peltekian, Muhammed Nur Talha Kilic, Wei-keng Liao, Alok Choudhary, and Ankit Agrawal. Simultaneously improving accuracy and computational cost under parametric constraints in materials property prediction tasks. Journal of Cheminformatics, 16(1):17, 2024.

34. Anthony Onwuli, Ashish V Hegde, Kevin VT Nguyen, Keith T Butler, and Aron Walsh. Element similarity in high-dimensional materials representations. Digital Discovery, 2(5):1558–1564, 2023.

35. Tim Hsu, Tuan Anh Pham, Nathan Keilbart, Stephen Weitzner, James Chapman, Penghao Xiao, S Roger Qiu, Xiao Chen, and Brandon C Wood. Efficient and interpretable graph network representation for angle-dependent properties applied to optical spectroscopy. npj Computational Materials, 8(1):151, 2022.

36. Yuxin You, Zhen Liu, Xiangchao Wen, Yongtao Zhang, Wei Ai, . Large language models meet graph neural networks: a perspective of graph mining. Mathematics, 10(7):1147, 2022.

37. Jongmin Han, Youngchun Kwon, Youn-Suk Choi, Seokho Kang, . Improving chemical reaction yield prediction using pre-trained graph neural networks. Journal of Cheminformatics, 16(1):25, 2024.

38. Vishu Gupta, Kamal Choudhary, Brian DeCost, Francesca Tavazza, Carelyn Campbell, Wei-keng Liao, Alok Choudhary, and Ankit Agrawal. Structure-aware graph neural network based deep transfer learning framework for enhanced predictive analytics on diverse materials datasets. npj Computational Materials, 10(1):1, 2024.

39. Zhenwen Sheng, Hui Zhu, Bo Shao, Yu He, Zhuang Liu, Suqin Wang, Ming Sheng, . Accelerated Discovery of Energy Materials via Graph Neural Network. Inorganics, 13(12):395, 2025.

40. Dennis Delali Kwesi Wayo. Ensembles of Graph and Physics-Informed Machine Learning for Scientific Modeling in Materials Science: A Review: DDK Wayo. Archives of Computational Methods in Engineering, 1–26, 2025.

41. Archit Anand, Priyanka Kumari, Ajay Kumar Kalyani. High throughput screening of new piezoelectric materials using graph machine learning and knowledge graph approach. Computational Materials Science, 246:113445, 2025.

42. Anubhav Jain, Shyue Ping Ong, Geoffroy Hautier, Wei Chen, William Davidson Richards, Stephen Dacek, Shreyas Cholia, Dan Gunter, David Skinner, Gerbrand Ceder, and Kristin a. Persson. The Materials Project: A materials genome approach to accelerating materials innovation. APL Materials, 1(1):011002, 2013.

43. Kamal Choudhary, Kevin F. Garrity, Andrew C. E. Reid, Brian DeCost, Adam J. Biacchi, Angela R. Hight Walker, Zachary Trautt, Jason Hattrick-Simpers, A. Gilad Kusne, Andrea Centrone, Albert Davydov, Jie Jiang, Ruth Pachter, Gowoon Cheon, Evan Reed, Ankit Agrawal, Xiaofeng Qian, Vinit Sharma, Houlong Zhuang, Sergei V. Kalinin, Bobby G. Sumpter, Ghanshyam Pilania, Pinar Acar, Subhasish Mandal, Kristjan Haule, David Vanderbilt, Karin Rabe, and Francesca Tavazza. JARVIS: An integrated infrastructure for data-driven materials design, 2020.

44. Daniel Wines, Ramya Gurunathan, Kevin F. Garrity, Brian DeCost, Adam J. Biacchi, Francesca Tavazza, and Kamal Choudhary. Recent progress in the JARVIS infrastructure for next-generation data-driven materials design. Applied Physics Reviews, 10(4):041302, 10 2023.

45. Chi Chen and Shyue Ping Ong. Atomsets as a hierarchical transfer learning framework for small and large materials datasets. npj Computational Materials, 7(1):173, 2021.

46. Jian-Gang Kong, Ke-Lin Zhao, Jian Li, Qing-Xu Li, Yu Liu, Rui Zhang, Jia-Ji Zhu, and Kai Chang. Self-supervised representations and node embedding graph neural networks for accurate and multi-scale analysis of materials. Machine Learning: Science and Technology, 5(3):035018, 2024.

47. Dipendra Jha, Logan Ward, Arindam Paul, Wei-keng Liao, Alok Choudhary, Chris Wolverton, and Ankit Agrawal. ElemNet: Deep learning the chemistry of materials from only elemental composition. Scientific reports, 8(1):17593, 2018.

570 48. Logan Ward, Ankit Agrawal, Alok Choudhary, and Christopher Wolverton. A General-
571 Purpose Machine Learning Framework for Predicting Properties of Inorganic Materials.
572 npj Computational Materials, 2(August):16028, 2016.

573 49. Vahe Tshitoyan, John Dagdelen, Leigh Weston, Alexander Dunn, Ziqin Rong, Olga
574 Kononova, Kristin A Persson, Gerbrand Ceder, and Anubhav Jain. Unsupervised
575 word embeddings capture latent knowledge from materials science literature. Nature,
576 571(7763):95–98, 2019.

577 50. Leigh Weston, Vahe Tshitoyan, John Dagdelen, Olga Kononova, Amalie Trewartha,
578 Kristin A Persson, Gerbrand Ceder, and Anubhav Jain. Named entity recognition and
579 normalization applied to large-scale information extraction from the materials science
580 literature. Journal of chemical information and modeling, 59(9):3692–3702, 2019.

581 51. Anthony Yu-Tung Wang, Steven K Kauwe, Ryan J Murdock, and Taylor D Sparks.
582 Compositionally restricted attention-based network for materials property predictions.
583 npj Computational Materials, 7(1):1–10, 2021.

584 52. Matthias Feurer, Aaron Klein, Katharina Eggensperger, Jost Springenberg, Manuel
585 Blum, and Frank Hutter. Efficient and robust automated machine learning. In Advances
586 in Neural Information Processing Systems 28 (2015), pages 2962–2970, 2015.

587 53. Kamal Choudhary, Brian DeCost, and Francesca Tavazza. Machine learning with force-
588 field-inspired descriptors for materials: Fast screening and mapping energy landscape.
589 Physical review materials, 2(8):083801, 2018.

590 54. Logan Ward, Ruoqian Liu, Amar Krishna, Vinay I Hegde, Ankit Agrawal, Alok Choud-
591 hary, and Chris Wolverton. Including crystal structure attributes in machine learning
592 models of formation energies via Voronoi tessellations. Physical Review B, 96(2):024104,
593 2017.

594 55. Dipendra Jha, Vishu Gupta, Logan Ward, Zijiang Yang, Christopher Wolverton, Ian
595 Foster, Wei-keng Liao, Alok Choudhary, and Ankit Agrawal. Enabling deeper learning
596 on big data for materials informatics applications. Scientific reports, 11(1):1–12, 2021.

597 56. Alfred Yan, Tatiana Sokolinski, William Lane, Jinwang Tan, Kim Ferris, and Emily M
598 Ryan. Applying transfer learning with convolutional neural networks to identify
599 novel electrolytes for metal air batteries. Computational and Theoretical Chemistry,
600 1205:113443, 2021.

601 57. Zhan Ma, Shu Wang, Minhee Kim, Kaibo Liu, Chun-Long Chen, and Wenxiao Pan.
602 Transfer learning of memory kernels for transferable coarse-graining of polymer dynam-
603 ics. Soft Matter, 17(24):5864–5877, 2021.

604 **Acknowledgements** This work was performed under the following financial assistance award
605 70NANB24H136 from U.S. Department of Commerce, National Institute of Standards and Tech-
606 nology as part of the Center for Hierarchical Materials Design (CHiMaD). Partial support is also
607 acknowledged from NSF award OAC-2331329 and Northwestern Center for Nanocombinatorics.
608 Certain commercial equipment, instruments, software, or materials are identified in this paper in
609 order to specify the experimental procedure adequately. Such identifications are not intended to

imply recommendation or endorsement by NIST, nor it is intended to imply that the materials or equipment identified are necessarily the best available for the purpose.

Author Contributions V.G. designed and carried out the implementation and experiments for the dynamic embedding under the guidance of A.A., A.C., and W.L.. Y.L. and M.N.T.K performed experiments to train some of the models and collect performance results. K.C. and D.W. provided the necessary domain expertise for this work. V.G., A.A., D.W., and K.C. wrote the manuscript. All authors discussed the results and reviewed the manuscript.

Competing Interests The authors declare that they have no competing interests.