


Learning Molecular Mixture Property Using Chemistry-Aware Graph Neural Network

Hengrui Zhang (张恒睿)¹, Tianxing Lai (来天行)², Jie Chen¹, Arumugam Manthiram²,
James M. Rondinelli^{3,*} and Wei Chen^{1,†}

¹Department of Mechanical Engineering, Northwestern University, Evanston, Illinois 60208, USA

²Materials Science and Engineering Program and Texas Materials Institute, The University of Texas at Austin, Austin, Texas 78712, USA

³Department of Materials Science and Engineering, Northwestern University, Evanston, Illinois 60208, USA

 (Received 12 March 2024; revised 28 April 2024; accepted 7 May 2024; published 12 June 2024)

Recent advances in machine learning (ML) are expediting materials discovery and design. One significant challenge facing ML for materials is the expansive combinatorial space of potential materials formed by diverse constituents and their flexible configurations. This complexity is particularly evident in molecular mixtures, a frequently explored space for materials, such as battery electrolytes. Owing to the complex structures of molecules and the sequence-independent nature of mixtures, conventional ML methods have difficulties in modeling such systems. Here, we present MolSets, a specialized ML model for molecular mixtures, to overcome the difficulties. Representing individual molecules as graphs and their mixture as a set, MolSets leverages a graph neural network and the deep sets architecture to extract information at the molecular level and aggregate it at the mixture level, thus addressing local complexity while retaining global flexibility. We demonstrate the efficacy of MolSets in predicting the conductivity of lithium battery electrolytes and highlight its benefits in the virtual screening of the combinatorial chemical space.

DOI: [10.1103/PRXEnergy.3.023006](https://doi.org/10.1103/PRXEnergy.3.023006)

I. INTRODUCTION

The design of materials and molecules requires an understanding of the structure–property relationships in a broad chemical space. Navigating the chemical space is challenging because the *combinatorial complexity*, originating from the diversity of constituents (e.g., atoms) and various configurations of the constituents (e.g., atomic arrangements), forms an expansive space with the number of candidates far exceeding the capability of experiments or computational simulations [1]. Despite the challenges, growing amounts of data have been collected experimentally and computationally in this pursuit. Machine learning (ML) methods can harness these data and efficiently establish valuable structure–property relationships [2]. ML has manifested potential in predictive modeling, virtual screening, as well as accelerating the design of novel materials and molecules [3–5].

For many materials systems of practical importance, however, the combinatorial complexity occurs not only at one level but at multiple levels. An example is *molecular mixtures*, a chemical space frequently explored in the search for electrolytes [6], coolants, and fuels [7], among others. Previously, ML models have been applied to physical properties in small subspaces [8,9], such as a binary liquid. The applicability of current ML models in predicting more complex properties within a broader space of molecular mixtures is limited by the multilevel combinatorial complexity. The arrangement of various atoms produces complexity at the molecular level, whereas the mixing of different molecules brings added complexity at the mixture level. The key challenge lies in *representation* [10], i.e., converting the structure into a digital format that the ML model can access and use. To accurately learn the structure–property relationships of molecular mixtures, it is crucial to find a meaningful representation that both (1) captures the relevant physical and chemical information and (2) reflects the similarity between data points.

Specifically, the challenge of representing mixtures and predicting their properties is threefold. First, for an individual molecule, the property is determined by its chemical composition and geometry, which should be encoded in the representation and exposed to the model. The atomic properties and molecular geometry form the

*jrondinelli@northwestern.edu

†weichen@northwestern.edu

Published by the American Physical Society under the terms of the [Creative Commons Attribution 4.0 International license](https://creativecommons.org/licenses/by/4.0/). Further distribution of this work must maintain attribution to the author(s) and the published article's title, journal citation, and DOI.

essential chemistry to be captured. Owing to their versatility in encoding this information, graphs have been widely adopted as a representation of molecules. Using the graph representation, graph neural networks (GNNs) have demonstrated efficacy in molecular modeling and design applications [11–14]. Second, between molecules, there are interactions that influence their structures and behaviors, making the mixture property nonadditive, which requires flexible methods that do not *a priori* assume a form of constituent interactions. Last, a random mixture is sequence-independent, i.e., (30% *A*, 70% *B*) is the same as (70% *B*, 30% *A*). In conventional ML models, which take vector inputs, the two ways of representing the same mixture will be viewed differently; this fails to reflect the similarity between data points and lacks robustness or efficiency [15]. To address this, the representation should possess *permutation invariance*, as “sets” in mathematics, and ML model architectures, such as deep sets [16,17], were developed for attaining permutation invariant modeling of sets.

Here, we represent a molecular mixture as a set of molecular graphs, encoding the “chemistry” (formulation and interactions) at both the molecular and mixture levels, and propose the MolSets ML model for predicting mixture properties from this representation. MolSets leverages (1) the GNN to extract information from molecular chemistry and geometry, (2) the attention mechanism [18] to learn the relative importance and interaction of

constituents, and (3) the deep sets architecture to ensure permutation invariance. We demonstrate the predictive power and interpretability of MolSets in modeling the ionic conductivity of molecular mixtures to facilitate the virtual screening of electrolytes for lithium-based batteries.

II. PROBLEM FORMULATION

The electrolyte is an essential component of various types of batteries [19,20], and a major family of electrolytes are solid, liquid, or gel mixtures of molecules or polymers [21–23]. Broadly, we may view these all as molecular mixtures, with a single-component molecule as a special case thereof. The design of electrolytes involves a range of metrics to be considered, including properties such as ionic conductivity, electrochemical stability window, chemical or electrochemical stability in contact with the cathode and anode, solvation structure, Coulombic efficiency, safety, cost, and sustainability. To demonstrate our proposed MolSets model, we choose the room temperature (298 K) ionic conductivity as a target property to be predicted from the molecular mixture formulation (constituents and their weight fractions). Figure 1 illustrates the problem formulation. Predicting conductivity serves as a preliminary step for the virtual screening of electrolytes, offering hints towards their actual performance and solvation structure. It should be noted that the MolSets model

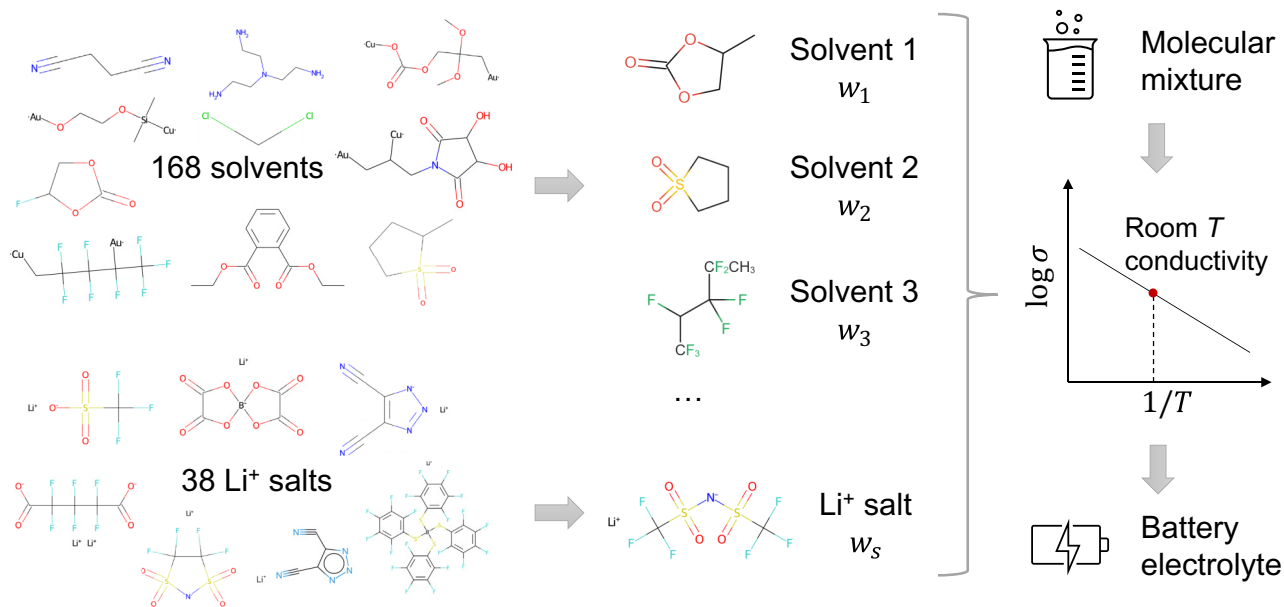


FIG. 1. Overview of the electrolyte conductivity prediction problem. We consider molecular mixtures consisting of 1–4 solvents and one Li⁺ salt in a candidate space of 168 types of solvents and 38 types of salts. Structures of some representative constituents are visualized. Solvents include both small molecules and polymers; for polymers, we show the structures of their monomers, where “Cu” and “Au” are placeholders to indicate the connecting sites of the monomer. (Cu and Au are not included in the molecular graph.) Our model is used to predict the room temperature conductivity of a molecular mixture as an indicator of its potential as a battery electrolyte.

is generally applicable to other properties or performance metrics as well.

We retrieved a dataset curated by Bradford *et al.* [6] from the previously published literature. The dataset reports the experimentally measured conductivity of polymer or molecular mixtures, each consisting of up to four types of molecules, together with Li^+ salts and inorganic additives. To avoid missing values, we select a subset of 1076 distinct mixtures, each of which is formulated of 1–4 different molecules and one salt, as a testbed for MolSets. Note that not all mixtures have conductivity reported at 298 K in the dataset, but they all have conductivities reported at multiple different temperatures. We use the reported ionic conductivities, σ , to perform a linear fit based on Arrhenius transport, which is widely used in modeling the conductivity of both small molecule and polymer mixtures [24,25]:

$$\log \sigma = -\frac{E_a}{RT} + \log \sigma_\infty = k\frac{1}{T} + b, \quad (1)$$

where the temperature dependence, k , is related to the activation energy, E_a , and the ideal gas constant, R . With the linear fit, we calculate an inferred 298 K conductivity for the mixtures without a reported value in the dataset.

III. MODEL DEVELOPMENT

A. Deep sets learning

A mixture with arbitrary constituents can be represented as a set, $X = \{x_1, x_2, \dots, x_m\}$, where each x is a distinct constituent, and the number of constituents, m , is not fixed. To model a mixture property, $y = f(X)$, permutation invariance should be ensured for any permutation of sequence π :

$$f(\{x_1, \dots, x_m\}) = f(\{x_{\pi(1)}, \dots, x_{\pi(m)}\}). \quad (2)$$

Without permutation invariance, an ML model may learn a false dependence of y on the sequence of constituents in X , thus lacking robustness to different ways of representing the same mixture. A model-agnostic solution is to include mixtures in various sequences in the training data. However, this leads to higher computational cost that grows significantly with the number of constituents in a mixture, thus lacking efficiency and limiting the scalability.

A more efficient and robust approach is to enforce permutation invariance by design of the model architecture. The deep sets architecture defines the sufficient and necessary principle for that:

$$f(X) = \rho\left(\sum_{x \in X} \phi(x)\right), \quad (3)$$

or, equivalently, $f(X) = \rho(\oplus_{x \in X} \{\phi(x)\})$, where $\phi(\cdot)$ and $\rho(\cdot)$ are appropriate transformations, and \oplus is any permutation invariant aggregating operation.

A unique feature of molecular mixtures is that the set representation becomes $X = \{(x_i, w_i)\}_{i=1}^m$, where each molecule x has a weight fraction w in the mixture. The permutation invariant model is modified accordingly:

$$f(X) = \rho(\oplus_{(x,w) \in X} \{\phi(x), w\}). \quad (4)$$

In our implementation, the model consists of three components: (1) an “embedding” module, ϕ , that learns a latent representation for each molecule; (2) an “aggregation” module, \oplus , that combines representations of individual molecules into a latent representation for the mixture; and (3) a “transformation” module, ρ , that maps the mixture representation to the target property. Figure 2(a) illustrates the overall model architecture.

B. Molecular graph neural network

We convert a molecule into the graph data structure, $G = (V, E)$, where the nodes V represent heavy atoms (any element except H) in the molecule, and edges E represent the bonds between them. Every node is associated with a list of features. To make it generalizable, we use only the element type and easily obtainable atomic descriptors for an atom: atomic mass, formal charge, electronegativity, van der Waals radius, and the number of H atoms connected to it. Every bond is associated with one feature, the bond type (aliphatic, aromatic, single, double, etc.). The atomic and bond descriptors are obtained using open-source cheminformatics software RDKit [26] and PYMATGEN [27]. In addition, to distinguish between small molecules and polymers, we associate every graph with the logarithmic molecular weight, $\log M$, as a graph-level feature.

We use a GNN to learn the molecular representations. As Fig. 2(b) shows, for an input graph, the GNN performs message passing between connected nodes using a “graph convolution” operator:

$$v'_i \leftarrow MP(v_i, v_j, e_{ij})_{j \in N(i)} \text{ for } v_i \in V. \quad (5)$$

In one convolutional layer, the feature vector of every node is updated using a “message” derived from features of its neighbor nodes and edges between them. How the message is derived varies, depending on the type of convolution operator. After several convolutional layers, the global mean value vector of all node features is calculated. This vector is concatenated with $\log M$ and goes into a fully connected layer, which generates a vector representation of the graph.

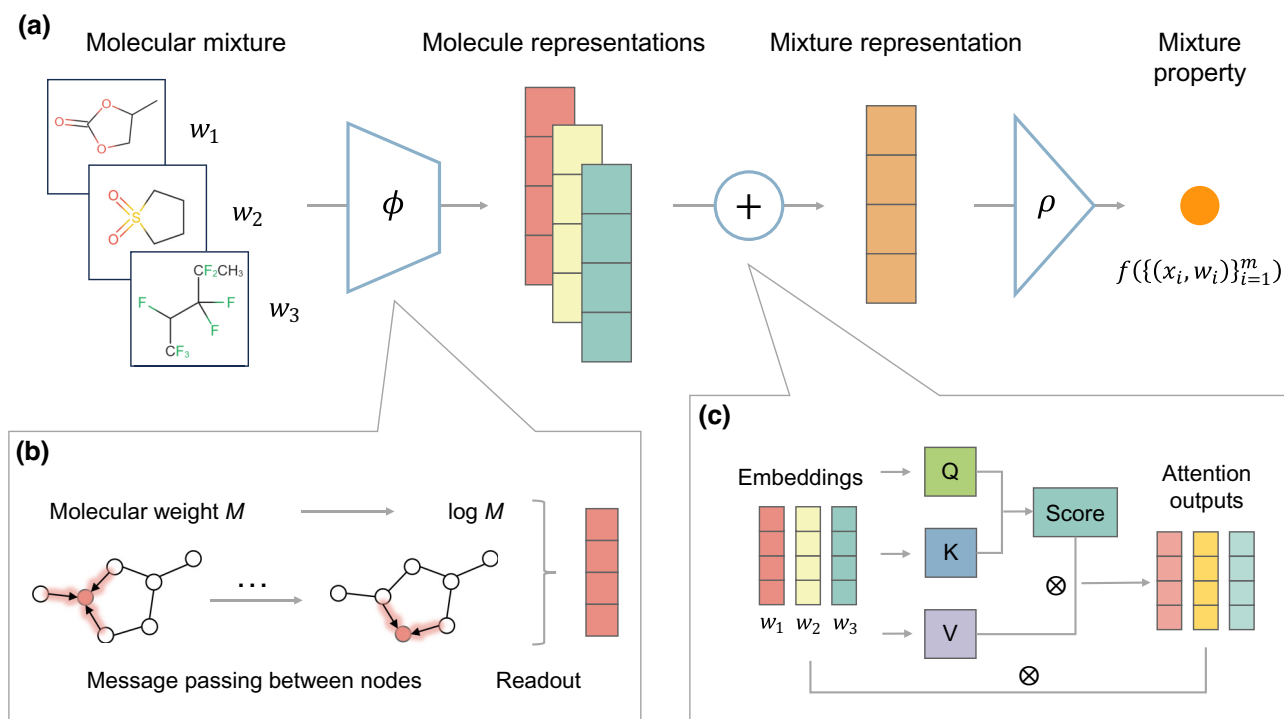


FIG. 2. (a) Architecture of MolSets. (b) Embedding module ϕ is a graph neural network, which performs iterative message passing between nodes and reads out all node features to form a representation of the molecule. (c) Aggregation module \oplus first uses the attention mechanism to adjust the molecular representations according to their importance, and then performs a weighted sum to form a mixture representation. Transformation module ρ is a multilayer perceptron composed of several fully connected layers that maps the mixture representation to the target property.

C. Aggregation with attention mechanism

The ϕ module maps every molecular graph x to a vector representation z . To aggregate the individual molecular representations and weight fractions $\{(z_i, w_i)\}_{i=1}^m$, an intuitive way is to perform a weighted summation. However, a constituent’s contribution to the mixture property is not solely linearly dependent on its weight fraction. We use the attention mechanism [18] to more accurately capture the constituents’ contributions. For each molecular representation z , a “query” vector q is calculated by $q = z \cdot W^Q$, with learnable parameters W^Q , and similarly two other vectors, “key” k and “value” v . Then, a “score” that reflects the molecule’s importance is derived from q and k , from which we get an updated molecular representation:

$$z' = \text{score} \cdot v = \text{softmax}\left(\frac{q \cdot k^T}{\sqrt{d_k}}\right) \cdot v, \quad (6)$$

where d_k is the length of k , and $\text{softmax}(\cdot)$ is a vector function:

$$\sigma(\mathbf{x})_i = \frac{\exp x_i}{\sum_{j=1}^n \exp x_j},$$

for any vector $\mathbf{x} = [x_1, \dots, x_n]$. The updated representation does not change in dimension, but its value is modulated according to its importance. In addition, permutation

invariance is maintained as the attention mechanism operates on each molecule separately with shared parameters.

Afterward, from the updated representations of molecules, the representation of the mixture is computed as a weighted sum: $Z = \sum_{i=1}^m w_i z'_i$. These form the aggregation module, \oplus , which maps an arbitrary number of molecular representation vectors into a single mixture representation, as shown in Fig. 2(c). The aggregation formulation accounts for the weight fractions of constituent molecules, as well as the nonadditive or nonlinear contributions of constituents on the mixture property.

D. MolSets model architecture

Based on the deep sets guidelines [Eq. (4)] and the two modules described above, we propose the MolSets model architecture, illustrated in Fig. 2. A molecular mixture is input as a set of graphs. The graphs are mapped by a GNN-based embedding module, ϕ , to representation vectors, which are then aggregated to a mixture representation vector by the \oplus module. Finally, the transformation module, ρ , a few layers of fully connected neural network, maps the mixture representation to the output, i.e., the target property of the mixture. Table I lists the detailed specifications of the MolSets model.

TABLE I. Implementation of three modules of MolSets and hyperparameters that could impact the model architecture.

Module	Implementation	Key hyperparameters
Embedding ϕ	Graph convolutional layers	Convolution operator type; number and dimensions of layers
	Global mean pooling	
	Fully connected layer	
Aggregation \oplus	Attention layer	Dimension of representation Dimensions of q, k, v
	Weighted sum	
	Fully connected layers	
Transformation ρ	Fully connected layers	Number of layers; dimensions of hidden layers

IV. RESULTS AND DISCUSSION

A. Model testing

As a demonstration of MolSets, we perform tests on the electrolyte conductivity prediction task described in Sec. II. Since an electrolyte contains two different types of molecules, solvents and salts, they can be treated either together or separately in the model. Here, we adopt a more general setting: constituents can be grouped into several categories, and a molecular mixture can be considered as mixtures within each category. In the case where no grouping is applicable, this reduces to the “all together” setting.

Following the setting, we adjust MolSets to a dual-pathway architecture to be compatible with the data: two ϕ modules are used, learning representations for solvents and salts, respectively. As each electrolyte consists of multiple solvents and only one salt in the dataset we use, the learned representations for solvents are aggregated into a solvent mixture representation using \oplus , while that for the salt needs no aggregation and is by itself the “salt mixture representation.” If multiple salts are considered, the salt mixture representation can be learned in the same way as solvents using ϕ and \oplus . Then, the solvent mixture representation, salt representation, and salt molality are concatenated and passed to ρ .

In the following part, we show the results of predicting electrolyte conductivity using MolSets and other methods. These tests serve three purposes: (1) as MolSets is a generic architecture that can work with various types of GNNs and has several hyperparameters, we explore the effect of some key configurations on its performance; (2) benchmarks that demonstrate MolSets’ advantages over existing models; and (3) ablation tests that investigate the importance of different parts of MolSets. We use two main metrics for quantifying the model performance. (1) The Pearson correlation coefficient, r_p , which measures the linear correlation between target and predicted values. It is the square root of another commonly used metric, coefficient of determination R^2 . (2) The Spearman rank correlation coefficient, r_s , which measures how well the predicted values are ranked correctly as the targets. It is important for materials modeling, especially for materials properties to be optimized, as correct ranking can guide the search towards superior candidates. For every test,

we randomly split the dataset into training, validation, and testing datasets in a 60%/20%/20% ratio. Since the combinations of models and configurations lead to many runs, cross-validation becomes time-consuming; hence, we use one fixed data split in the model selection and hyperparameter tuning. For every different configuration of MolSets, hyperparameter tuning is performed using training and validation data, and performance metrics are assessed on the testing data as the criteria of selecting the optimal configuration. Then, a comparison of MolSets and other models is conducted on 30 random splits of the dataset.

B. Benchmark and ablation test

A major customizable configuration of MolSets is the type of GNN used as the ϕ module. An abundance of GNNs have been developed, and their main difference lies in the graph convolution operation, i.e., how messages are composed and passed between nodes. Out of many available, we choose four commonly used general-purpose graph convolution operators, GraphConv, SAGEConv, GCNConv, and GATConv, as well as a directed message passing scheme (DMPNN) specially designed for molecules [28]. Integrating each convolution operator to MolSets, we tune the key hyperparameters that have an impact on the model architecture using the validation dataset and assess the testing performances via r_p and r_s . The models are implemented using the PyTorch Geometric library [29], and more details on model implementation are presented in Appendix B. The performance of MolSets integrated with different graph convolution operators is presented in Table S2 within the Supplemental Material [49]. Using GraphConv and SAGEConv, both generic and lightweight convolution operators, MolSets attain high regression and ranking accuracies on the testing data. DMPNN also shows high testing accuracy; although specially designed for molecules, the performance of DMPNN is not superior, possibly because its architecture is not tailored for working with the \oplus and ρ modules as a part of MolSets. Based on this step, we choose GraphConv as the convolution operator in MolSets for further tests.

Next, we train two different GNN-based deep learning models to predict mixture properties from the constituents’ molecular graphs. (1) Replace the \oplus aggregation module of MolSets with a simple weighted summation. This model

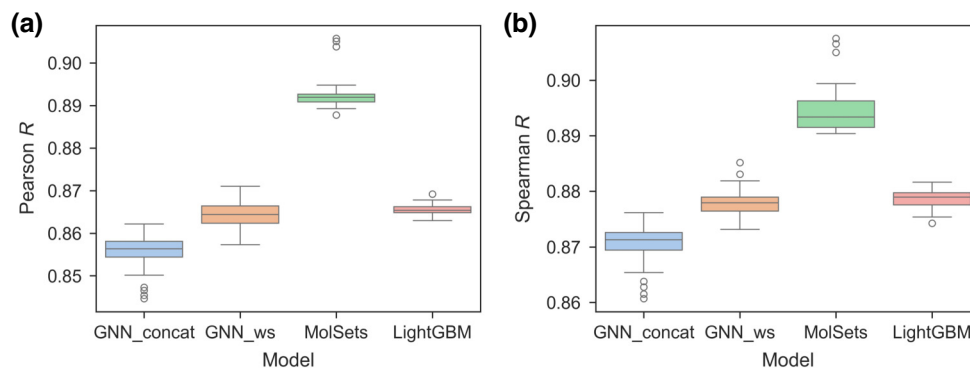


FIG. 3. Performance metrics of MolSets and other models on the testing dataset: (a) Pearson correlation coefficient, r_p , and (b) Spearman rank correlation coefficient, r_s , across 30 random data splits.

still satisfies Eq. (3) and is thus permutation invariant. However, it models a mixture as a linear combination of its constituents, ignoring the nonlinear contributions they may have. (2) Instead of aggregation, the molecular representation vectors learned by a GNN ϕ are concatenated and padded with zeroes, together with the solvent weight fractions and salt molality, to form a mixture representation vector, which then goes through ρ . This model no longer preserves permutation invariance and needs the maximum number of constituents to be specified. To control variables, other hyperparameters are kept the same as those used in the MolSets model with GraphConv. The performance of MolSets and other models is summarized in Fig. 3, from which we find that MolSets shows a systematic advantage in learning both correlation (r_p) and ranking (r_s) compared to existing GNNs.

As another benchmark, we retrieve a list of numerical molecular descriptors and predict the conductivity based on these descriptors using gradient boosting. 208 descriptors are calculated for every molecule using RDKit. Then, for every mixture, the descriptors of its constituents (4 solvents and 1 salt) together with their logarithmic molecular weights and weight fractions are concatenated to form a feature vector. For mixtures containing fewer than four solvents, the positions corresponding to missing solvents are filled with zeros. The dimensions containing abnormal values (NaN or infinity) or with no variance across datapoints are removed. LightGBM [30], an accurate and efficient implementation of the gradient boosting decision tree algorithm, is employed to fit a model using the featurized data, and its performance across 30 replicates is shown in Fig. 3. Like the GNN with concatenation, this method is also not permutation invariant and restricted to a prespecified number of constituents, and it does not attain an accuracy as high as that of MolSets.

In Fig. 4, we show the true values versus model-predicted values of logarithm conductivity on the testing data for one split, with colors differentiating different types of mixtures. In general, MolSets' predictions show less

deviation from the true values; moreover, MolSets models show lower errors at the high end and low end of conductivity (points far from the center), which favors the discovery of exceptional materials [31].

C. Analyses and interpretation

The fundamental uniqueness of MolSets, compared to other ML models, is that it treats a mixture as a permutation invariant set of constituents, the contributions of which are not simply governed by their weight fractions. The advantage of this assumption is demonstrated by the better predictive performance. To interpret why this assumption can lead to better performance, we investigate two questions. (1) How is a mixture different from the weighted summation of its constituents? (2) What happens if a mixture model is not permutation invariant?

For (1), we probe the representation space learned by the ϕ and \oplus modules of MolSets. Every molecular mixture input to MolSets with GraphConv as a set of graphs is mapped to a 32-dimensional representation vector. Note that a “mixture” with only one constituent can be viewed as a representation of that constituent. Choosing three small molecules, together with one binary mixture and one ternary mixture among them, we investigate their locations in the representation space. We use t -distributed stochastic neighbor embedding (t -SNE) [32] to reduce the 32-dimensional vectors to 2 dimensions and visualize the locations of constituents, mixtures (marked by text), and weighted summations of constituents (indicated by arrows) in the representation space in Fig. 5(a). The significant deviation of the mixture representation from the weighted summations (dashed lines) suggests that in MolSets' representation learning, the mixture is not formed as a linear combination of its constituents; this highlights MolSets' capability to capture the nonlinearity of a mixture.

For (2), we test the performance of the permutation noninvariant GNN model (concatenating molecular representations) on a dataset containing mixtures represented in

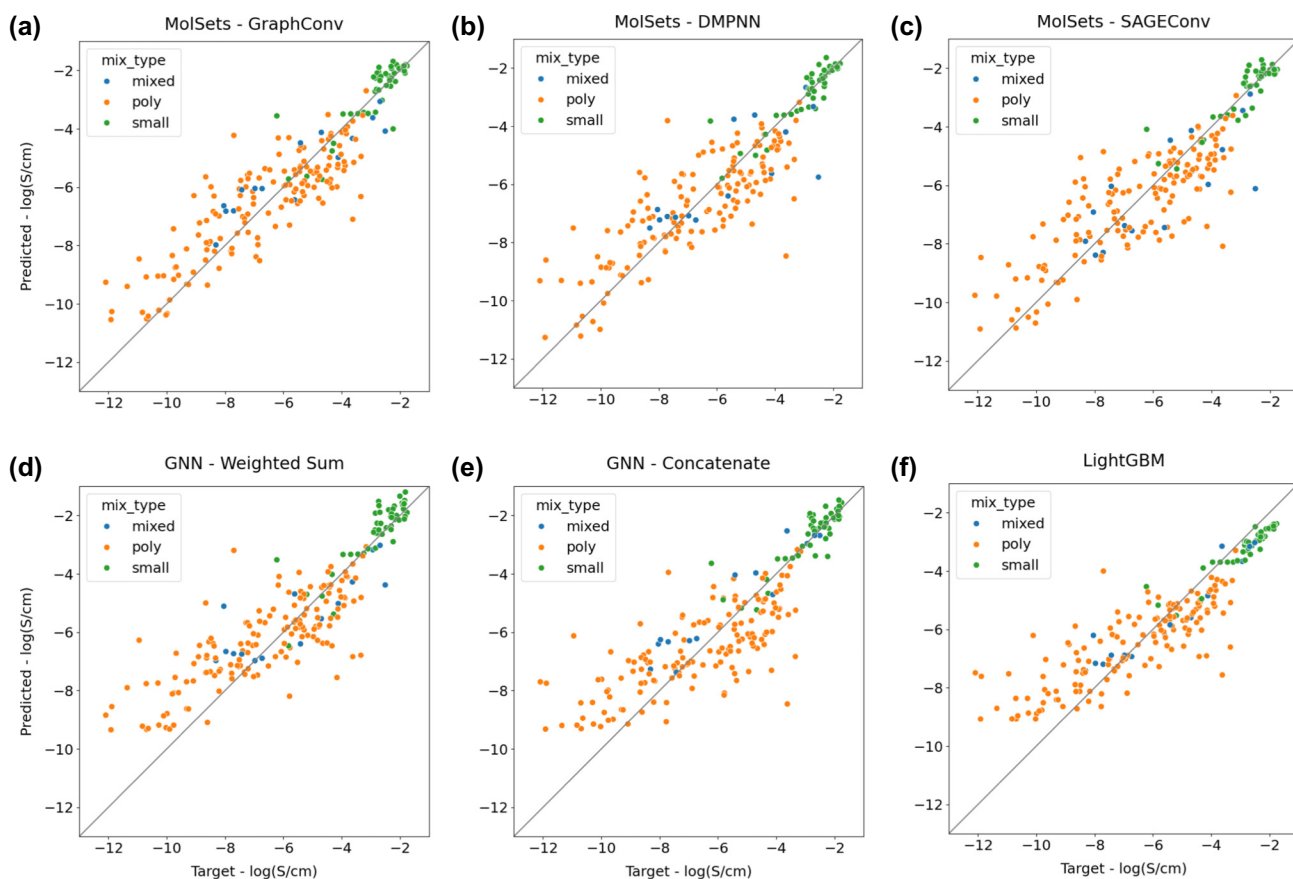


FIG. 4. Regression plots showing target values (horizontal axis) and predicted values (vertical axis) of logarithmic conductivity. (a)–(c) MolSets using different graph convolution operators. (d)–(e) GNN models using the GraphConv operator and different treatments of molecular representations learned from graphs. (f) LightGBM model trained on molecular descriptors. “Mix_type” denotes whether a data point is a mixture of molecules (“small”), polymers (“poly”), or both (“mixed”).

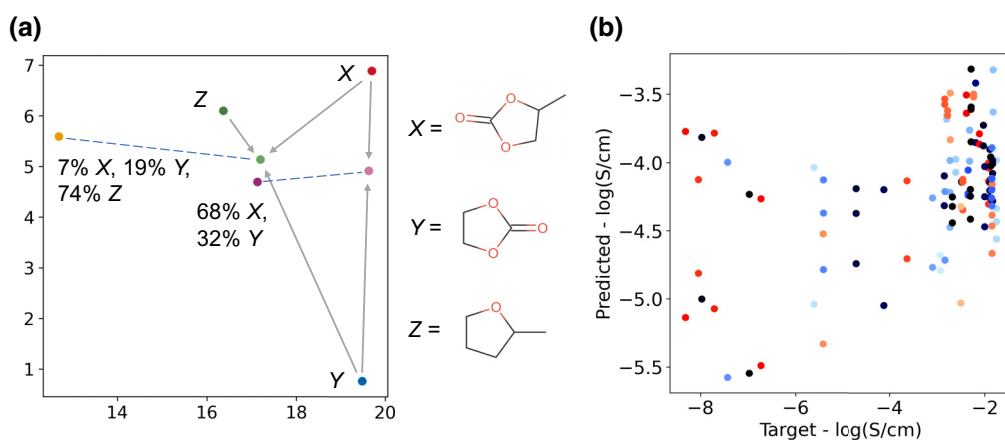


FIG. 5. (a) Reduced-dimensional visualization of three molecules (shown on the right) and their binary or ternary mixtures in the learned representation space. Arrows indicate the weighted summation of constituents' representations, the deviation of which from the learned representations of mixtures is indicated by dashed lines. (b) Target values (horizontal axis) and GNN-predicted values (vertical axis) on the permuted dataset. Colors are assigned according to target values to help distinguish different mixtures.

different sequences. From the testing dataset, we select all the mixtures that contain more than one solvent and create a new dataset where the solvents in every mixture appear in different orders. For each permuted data point, the order of solvents, molecular weights, and weight fractions are permuted in the same way; hence, it represents the same mixture, and the target property is unchanged. Figure 5(b) shows the GNN model's predictions on this permuted dataset: it gives substantially different predictions for the same mixture represented in different orders. Moreover, the model's predictions show systematic deviation from the target values on these mixtures containing more than one constituent. These demonstrate the advantages of permutation invariance in mixture modeling.

With an accurate predictive model like MolSets, it is also desired to draw chemical insights that guide theoretical or experimental studies. GNNs and attention models are both challenging to interpret due to their complexity. As a preliminary step, we inspect the importance of constituents in small molecule mixtures. We use the magnitude change of representation vector $\|z'\|/\|z\|$ as an indicator of relative importance for each constituent (details are provided in Appendix B). Most molecules showing high importance are cyclic carbonate esters, which are widely used in electrolytes for their superior ion solvation ability [33]. We make the relative importance data available in the Supplemental Material for further analyses, e.g., the interaction between different types of molecules in the mixture.

D. Virtual screening and experimental assessment

Finally, we employ the MolSets model with the best configurations found in tests to perform virtual screening. We train the model on the whole dataset (70% for training, 30% for validation), and use it to predict the

room temperature conductivity for a large set of candidates with the small molecules and Li^+ salts that appear in at least three mixtures in the dataset. We consider all equal-weight binary mixtures among 28 types of small molecules, combining with 30 types of salts (1 mol kg^{-1}), totaling 11 340 candidates. The predicted conductivities of all candidates are available at Ref. [34]. Out of the candidates, the top-performing ones are listed in Table S4 within the Supplemental Material [49]. Fixing the type and molality of salt, we compare the equal-weight binary mixtures with their constituents and find that a binary mixture can lead to higher conductivity than either of its constituent molecules (Table S5 within the Supplemental Material [49]). This indicates a nontrivial improvement in the properties of mixtures, which resembles the *bowing effect* observed in alloys [35], and provides the potential to design electrolytes with superior properties to those of single constituents within the broader chemical space of mixtures.

In addition to the candidates selected based on the available dataset, we choose a few new mixtures that are of special interest to small-molecule electrolyte developers, and experimentally assess the prediction of MolSets. Each candidate is a binary or ternary equimolar mixture with 1 mol kg^{-1} lithium bis(trifluoromethanesulfonyl)imide (LiTFSI) added. Note that these mixtures contain molecules never seen by the model, and they have high similarities in composition, both making the task more challenging. The mixtures were prepared inside a glovebox and measured with a conductivity probe at 298 K (details are provided in Appendix C). Figure 6 shows the results, where MolSets' predictions show fair agreement with experimentally measured conductivities. Assessment results for other ML models are shown in Fig. S3 within the Supplemental Material [49]. Were there a specially curated dataset for small molecular

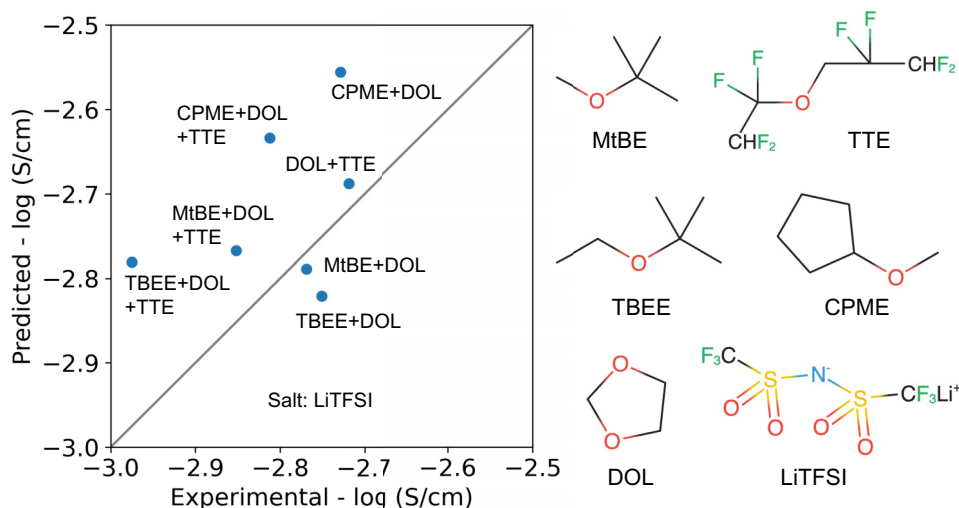


FIG. 6. Predicted versus experimentally measured logarithm conductivities of mixtures, with the ingredient structures shown on the right.

mixtures using high-throughput experiments [36,37], the capability of MolSets on these virtual screening tasks could be further improved.

A limitation of the current model is that it does not consider salt solubility. Since the dataset is curated from experimental reports in the literature, where salt molality does not exceed solubility, the model trained thereon cannot generalize to mixtures containing higher salt concentrations. Nonetheless, by rationally choosing salt molality as a constraint, this issue can be precluded when applying MolSets to electrolyte screening.

V. SUMMARY AND OUTLOOK

We presented MolSets, a machine learning model architecture for molecular mixtures that captures the chemistry and geometry of molecules while preserving the permutation invariance nature of mixtures. Using the conductivity of electrolytes as a testbed, we demonstrated the accuracy and robustness of MolSets, and investigated the nontrivial characteristics of mixtures compared to the combination of constituents.

An accurate yet efficient predictive model like MolSets can facilitate the virtual screening of promising materials in the vast combinatorial space of molecular mixtures. As an initial step, we use MolSets to predict the conductivity of over 10 000 mixtures based on available data. However, the limited availability of data poses a key challenge. Although a molecular property database was released recently [38], there is no comprehensive data resource for mixtures. As datasets for various properties of mixtures become available, the generalizability of MolSets can be further examined. A future direction could be constructing a platform where researchers could access MolSets-predicted properties of mixtures and upload experimentally measured values. With growing amounts of data and the MolSets model, such a platform could offer accurate estimates of molecular mixture properties, as AlphaFold [39] offers for proteins and matterverse.ai [40] for crystals.

Raw data are retrieved from Ref. [6]. The code and processed data are publicly available at GitHub [50]. The newly generated data are available at Dryad [34].

ACKNOWLEDGMENTS

This modeling work at Northwestern University was supported in part by the National Science Foundation, Division of Materials Research, under Awards No. 2324173 and No. 2219489. H.Z. received support from the Ryan Graduate Fellowship. The experimental work at UT Austin was supported by the U.S. Department of Energy, Office of Basic Energy Sciences, Division of Materials Science and Engineering under Award No. DE-SC0005397. The authors thank Liwei Wang, Akash Pandey, Xiaoyu

Xie, Yaxin Cui, Daniel Apley, Guilherme Missaka, Jeffrey Lopez, Yuxin Chang, and Edward Sargent for helpful discussions.

H.Z. conceived the project, developed the methods, conducted the tests, and drafted the manuscript. T.L. conducted the experiments. J.C. contributed to method development and analysis. A.M., J.M.R., and W.C. supervised the project. T.L. conducted the experiments. All authors reviewed and revised the manuscript.

APPENDIX A: DATA PREPARATION

We use a dataset of size 1076, involving both small molecules and polymers, due to limited available data for mixture properties. Specifically, 732 data points are polymer mixtures, 269 are small molecule mixtures, and 75 contain both small molecule(s) and polymer(s). We represent a polymer with the molecular graph of its monomer; as a monomer is linked to another in the real polymer, we add C atoms to its connecting sites in the graph. In addition, we include the molecular weight, M , in the input, so that polymers can be distinguished from small molecules. In the dataset, some polymers have the weight average molecular weight, M_w , reported and some have the number average molecular weight, M_n . We use the available one as M and use M_n if both are available.

A molecule is converted into a graph with its heavy atoms (any element other than H) as nodes and bonds between them as edges. The node features are 13-dimensional, with the first 7 dimensions being one-hot encoding of elements B, C, N, O, F, S, and Cl. If the atom is one of these elements, the corresponding dimension is 1 and others are 0; otherwise, these seven dimensions are all 0. The remaining six dimensions are atomic number, atomic mass (in Da), formal charge (in elementary charge e), electronegativity (on the Pauling scale), van der Waals radius (in Å), and number of H atoms connected to the atom. Each edge is associated with one feature, a numerical representation of the bond type: 1, 2, and 3 for single, double, and triple bonds, respectively, and 1.5 for aromatic bonds.

APPENDIX B: METHODS AND MODEL

1. Graph convolution

A graph is formulated as $G = (V, E)$, with nodes $V = \{v_i\}_{i=1}^n$ and edges $E = \{e_{ij}\}$. Each node v_i is associated with a node feature vector x_i , and in our formulation, each edge is associated with a scalar feature, denoted e_{ij} for simplicity. Graph convolution iteratively updates the node feature vectors. A graph convolution operator defines (1) how a “message” is composed of a node v_i , neighbor nodes $j \in N(i)$, and edges e_{ij} between them, and (2) how the node’s feature is updated based on the message.

TABLE II. Definitions of graph convolution operators.

Convolution operator	Message passing scheme
SAGEConv [41]	$x'_i = W_1 x_i + W_2 \cdot \text{mean}_{j \in N(i)} x_j$
GraphConv [42]	$x'_i = W_1 x_i + W_2 \sum_{j \in N(i)} e_{ij} x_j$
GCNConv [43]	$x'_i = W^T \sum_{j \in N(i) \cup \{i\}} \left(e_{ij} / \sqrt{d_i d_j} \right) x_j$
GATConv [44]	$x'_i = \alpha_{ij} W_1 x_i + \sum_{j \in N(i)} \alpha_{ij} W_2 x_j$

Table II shows the graph convolution operators used here and their message passing schemes. In the formulas, W 's denote learnable weight matrices; GCNConv utilizes the quantity $d_i = 1 + \sum_{j \in N(i)} e_{ij}$; GATConv involves pairwise attention coefficients α_{ij} . Weight matrices are learned from and used for all nodes. More details are provided in the online documentation of PyTorch Geometric [29].

2. Model training and hyperparameter tuning

The model is trained using an AdamW optimizer [45], with an initial learning rate of 0.001 and a weight decay coefficient of 0.0001 for regularization. The learning rate is controlled using a scheduler, which reduces it by a factor of 2 upon 10 epochs of no improvement in validation loss. To further prevent overfitting, an early stopping rule is adopted in training: if validation loss has not improved for 20 successive epochs, the training process is terminated and the model parameters at the epoch that displayed the lowest validation loss are taken as the final model parameters.

Implementing MolSets with each graph convolution operator, we tune the hyperparameters that have an impact on model architecture, which include (1) the number of convolution layers, (2) the dimension of hidden layers, (3) the dimension of learned molecular representation, and (4) the dimension of attention. Tuning is conducted using a grid search over a discrete set of values for each hyperparameter and tracked using the Weights & Biases platform [46]. Table S1 within the Supplemental Material [49] lists the hyperparameters found to be optimal for MolSets with different convolution operators. Figure S1 within the Supplemental Material [49] presents a typical learning curve of MolSets with GraphConv; other models show similar learning curves.

3. Analysis and interpretation

The \oplus module of MolSets is designed to learn the importance of constituents in a mixture using the attention mechanism. However, interpreting an attention model to retrieve the importance scores is highly nontrivial [47,48].

Here, we adopt an intuitive method, which is not necessarily rigorous. A molecule is mapped to its representation vector z by the GNN module ϕ , and the attention mechanism converts it into z' before forming a mixture representation. We compute $\|z\|_2$, the Euclidean norm of z , as well as $\|z'\|_2$. The norm of a vector quantifies its magnitude, and intuitively, the attention mechanism should enlarge the magnitude of an important constituent's representation vector. Hence, we use $\|z'\|_2/\|z\|_2$ as an indicator of relative importance.

In every mixture, we find constituents with an importance that is more than twice that of the least important constituent. These constituents include ethylene carbonate [O=C1OCCO1], propylene carbonate [CC1COC(=O)O1], and γ -butyrolactone [O=C1CCCCO1], which are cyclic carbonate esters, and toluene [CC1=CC=CC=C1].

APPENDIX C: EXPERIMENTAL DETAILS

1,3-Dioxolane (DOL, Thermo Scientific), 1,1,2,2-tetrafluoroethyl-2,2,3,3-tetrafluoropropylether (TTE, SynQuest), methyl *tert*-butyl ether (MtBE, Thermo Scientific), *tert*-butyl ethyl ether (TBEE, Sigma-Aldrich), and cyclopentyl methyl ether (CPME, Thermo Scientific) were dried with molecular sieves inside an argon-filled glovebox ($O_2, H_2O < 0.1$ ppm). To prepare the electrolytes, 1 mol kg⁻¹ LiTFSI (Solvionic) was dissolved in various solvent mixtures. The conductivities of the electrolytes were measured with a conductivity probe (InLab 752, Mettler Toledo) at 25 °C inside the glovebox. Table S6 within the Supplemental Material [49] lists the mixtures' constituents and their predicted and measured conductivities. No unexpected or unusually high safety hazards were encountered.

-
- [1] T. Lombardo *et al.*, Artificial intelligence applied to battery research: Hype or reality?, *Chem. Rev.* **122**, 10899 (2022).
 - [2] K. T. Butler, D. W. Davies, H. Cartwright, O. Isayev, and A. Walsh, Machine learning for molecular and materials science, *Nature* **559**, 547 (2018).
 - [3] K. M. Jablonka, D. Ongari, S. M. Moosavi, and B. Smit, Big-data science in porous materials: Materials genomics and machine learning, *Chem. Rev.* **120**, 8066 (2020).
 - [4] N. Fedik *et al.*, Extending machine learning beyond interatomic potentials for predicting molecular properties, *Nat. Rev. Chem.* **6**, 653 (2022).
 - [5] Y.-C. Gao, N. Yao, X. Chen, L. Yu, R. Zhang, and Q. Zhang, Data-driven insight into the reductive stability of ion-solvent complexes in lithium battery electrolytes, *J. Am. Chem. Soc.* **145**, 23764 (2023).
 - [6] G. Bradford, J. Lopez, J. Ruza, M. A. Stolberg, R. Osterude, J. A. Johnson, R. Gomez-Bombarelli, and Y. Shao-Horn, Chemistry-informed machine learning for polymer electrolyte discovery, *ACS Cent. Sci.* **9**, 206 (2023).

- [7] N. Kuzhagaliyeva, S. Horváth, J. Williams, A. Nicolle, and S. M. Sarathy, Artificial intelligence-driven design of fuel mixtures, *Commun. Chem.* **5**, 111 (2022).
- [8] S. Ajmani, S. C. Rogers, M. H. Barley, and D. J. Livingstone, Application of QSPR to mixtures, *J. Chem. Inf. Model.* **46**, 2043 (2006).
- [9] I. Oprisiu, E. Varlamova, E. Muratov, A. Artemenko, G. Marcou, P. Polishchuk, V. Kuz'min, and A. Varnek, QSPR approach to predict nonadditive properties of mixtures. Application to bubble point temperatures of binary mixtures of liquids, *Mol. Inf.* **31**, 491 (2012).
- [10] J. Damewood, J. Karaguesian, J. R. Lunger, A. R. Tan, M. Xie, J. Peng, and R. Gómez-Bombarelli, Representations of materials for machine learning, *Annu. Rev. Mater. Res.* **53**, 399 (2023).
- [11] J. Gilmer, S. S. Schoenholz, P. F. Riley, O. Vinyals, and G. E. Dahl, in *International Conference on Machine Learning* (PMLR, 2017), pp. 1263, <https://proceedings.mlr.press/v70/gilmer17a.html>.
- [12] G. Johannes, G. Janek, and G. Stephan, in *International Conference on Learning Representations* (ICLR, 2020), <https://openreview.net/forum?id=B1eWbxStPH>.
- [13] Y. Wang, J. Wang, Z. Cao, and A. Barati Farimani, Molecular contrastive learning of representations via graph neural networks, *Nat. Mach. Intell.* **4**, 279 (2022).
- [14] V. Sharma, M. Giammona, D. Zubarev, A. Tek, K. Nugyuen, L. Sundberg, D. Congiu, and Y.-H. La, Formulation graphs for mapping structure-composition of battery electrolytes to device performance, *J. Chem. Inf. Model.* **63**, 6998 (2023).
- [15] J. Westermayr, M. Gastegger, K. T. Schütt, and R. J. Maurer, Perspective on integrating machine learning into computational chemistry and materials science, *J. Chem. Phys.* **154**, 230903 (2021).
- [16] M. Zaheer, S. Kottur, S. Ravanbakhsh, B. Póczos, R. R. Salakhutdinov, and A. J. Smola, in *Neural Information Processing Systems* (Curran Associates, Inc., 2017), https://papers.nips.cc/paper_files/paper/2017/hash/f22e4747da1aa27e363d86d40ff442fe-Abstract.html.
- [17] J. Lee, Y. Lee, J. Kim, A. Kosiosek, S. Choi, and Y. W. Teh, in *International Conference on Machine Learning* (PMLR, 2019), pp. 3744, <https://proceedings.mlr.press/v97/lee19d.html>.
- [18] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, Ł Kaiser, and I. Polosukhin, in *Neural Information Processing Systems* (Curran Associates, Inc., 2017), https://proceedings.neurips.cc/paper_files/paper/2017/hash/3f5ee243547dee91fbd053c1c4a845aa-Abstract.html.
- [19] Q. Wang, L. Jiang, Y. Yu, and J. Sun, Progress of enhancing the safety of lithium ion battery from the electrolyte aspect, *Nano Energy* **55**, 93 (2019).
- [20] H. Wan, J. Xu, and C. Wang, Designing electrolytes and interphases for high-energy lithium batteries, *Nat. Rev. Chem.* **8**, 30 (2023).
- [21] A. Manthiram, X. Yu, and S. Wang, Lithium battery chemistries enabled by solid-state electrolytes, *Nat. Rev. Mater.* **2**, 16103 (2017).
- [22] J. Lopez, D. G. Mackanic, Y. Cui, and Z. Bao, Designing polymers for advanced battery chemistries, *Nat. Rev. Mater.* **4**, 312 (2019).
- [23] S. C. Kim *et al.*, Data-driven electrolyte design for lithium metal anodes, *Proc. Natl. Acad. Sci. U. S. A.* **120**, e2214357120 (2023).
- [24] F. Rahmanian, M. Vogler, C. Wölke, P. Yan, S. Fuchs, M. Winter, I. Cekic-Laskovic, and H. S. Stein, Conductivity experiments for electrolyte formulations and their automated analysis, *Sci. Data* **10**, 43 (2023).
- [25] V. Bocharova and A. P. Sokolov, Perspectives for polymer electrolytes: A view from fundamentals of ionic conductivity, *Macromolecules* **53**, 4141 (2020).
- [26] RDKit: Open-source cheminformatics. <https://www.rdkit.org>.
- [27] S. P. Ong *et al.*, Python Materials Genomics (PYMATGEN): A robust, open-source PYTHON library for materials analysis, *Comput. Mater. Sci.* **68**, 314 (2013).
- [28] K. Yang *et al.*, Analyzing learned molecular representations for property prediction, *J. Chem. Inf. Model.* **59**, 3370 (2019).
- [29] M. Fey and J. E. Lenssen, in *ICLR Workshop on Representation Learning on Graphs and Manifolds* (ICLR, 2019), <https://arxiv.org/abs/1903.02428>.
- [30] G. Ke, Q. Meng, T. Finley, T. Wang, W. Chen, W. Ma, Q. Ye, and T.-Y. Liu, in *Neural Information Processing Systems* (Curran Associates, Inc., 2017), https://papers.nips.cc/paper_files/paper/2017/hash/6449f44a102fde848669bdd9eb6b76fa-Abstract.html.
- [31] J. Schrier, A. J. Norquist, T. Buonassisi, and J. Brgoch, In pursuit of the exceptional: Research directions for machine learning in chemical and materials science, *J. Am. Chem. Soc.* **145**, 21699 (2023).
- [32] L. Van der Maaten and G. Hinton, Visualizing Data using t-SNE, *J. Mach. Learn. Res.* **9**, 2579 (2008), <https://jmlr.org/papers/v9/vandermaaten08a.html>.
- [33] C.-C. Su, M. He, R. Amine, T. Rojas, L. Cheng, A. T. Ngo, and K. Amine, Solvating power series of electrolyte solvents for lithium batteries, *Energy Environ. Sci.* **12**, 1249 (2019).
- [34] <https://doi.org/10.5061/dryad.pnvx0k6w1>.
- [35] A. R. Denton and N. W. Ashcroft, Vegard's law, *Phys. Rev. A* **43**, 3161 (1991).
- [36] C. W. Coley *et al.*, A robotic platform for flow synthesis of organic compounds informed by AI planning, *Science* **365**, eaax1566 (2019).
- [37] A. A. Volk, R. W. Epps, D. T. Yonemoto, B. S. Masters, F. N. Castellano, K. G. Reyes, and M. Abolhasani, AlphaFlow: Autonomous discovery and optimization of multi-step chemistry using a self-driven fluidic lab guided by reinforcement learning, *Nat. Commun.* **14**, 1403 (2023).
- [38] E. W. C. Spotte-Smith *et al.*, A database of molecular properties integrated in the Materials Project, *Digital Discovery* **2**, 1862 (2023).
- [39] J. Jumper *et al.*, Highly accurate protein structure prediction with AlphaFold, *Nature* **596**, 583 (2021).
- [40] C. Chen and S. P. Ong, A universal graph deep learning interatomic potential for the periodic table, *Nat. Comput. Sci.* **2**, 718 (2022).
- [41] W. L. Hamilton, R. Ying, and J. Leskovec, in *Neural Information Processing Systems* (Curran Associates, Inc., 2017), https://papers.nips.cc/paper_files/paper/2017/hash/5dd9db5e033da9c6fb5ba83c7a7e9bea9-Abstract.html.

- [42] C. Morris, M. Ritzert, M. Fey, W. L. Hamilton, J. E. Lenssen, G. Rattan, and M. Grohe, in *Proceedings of the AAAI conference on artificial intelligence* (AAAI Press, 2019), pp. 4602.
- [43] T. N. Kipf and M. Welling, in *International Conference on Learning Representations* (ICLR, 2017), <https://openreview.net/forum?id=SJU4ayYgl>.
- [44] P. Veličković, G. Cucurull, A. Casanova, A. Romero, P. Liò, and Y. Bengio, in *International Conference on Learning Representations* (ICLR, 2018), <https://openreview.net/forum?id=rJXMpikCZ>.
- [45] I. Loshchilo and F. Hutter, in *International Conference on Learning Representations* (ICLR, 2019), <https://openreview.net/forum?id=Bkg6RiCqY7>.
- [46] L. Biewald, Experiment Tracking with Weights and Biases, 2020. <https://www.wandb.com/>.
- [47] S. Serrano and N. A. Smith, in *Annual Meeting of the Association for Computational Linguistics (ACL, Florence, Italy, 2019)*, pp. 2931.
- [48] B. Bai, J. Liang, G. Zhang, H. Li, K. Bai, and F. Wang, in *KDD: Knowledge Discovery and Data Mining (ACM, Virtual Event, Singapore, 2021)*, pp. 25.
- [49] See the Supplemental Material at <http://link.aps.org/supplemental/10.1103/PRXEnergy.3.023006> for additional details and results of model training, performances, and virtual screening, as well as the relative importance of constituents in mixtures derived from model interpretation.
- [50] <https://github.com/Henrium/MolSets>.