Physics-constrained Automatic Feature Engineering for Predictive Modeling in Materials Science

Ziyu Xiang[1], Mingzhou Fan[1], Guillermo Vázquez Tovar[2], William Trehem[2], Byung-Jun Yoon[1,4], Xiaofeng Qian[2], Raymundo Arroyave[2], Xiaoning Qian[1,3]

[1]Electrical & Computer Engineering, [2]Materials Science & Engineering, [3]Computer Science & Engineering,
Texas A&M University, College Station, Texas 77843,
[4]Computational Science Initiative, Brookhaven National Laboratory, Upton, NY 11973

Abstract

Automatic Feature Engineering (AFE) aims to extract useful knowledge for interpretable predictions given data for the machine learning tasks. Here, we develop AFE to extract dependency relationships that can be interpreted with functional formulas to discover physics meaning or new hypotheses for the problems of interest. We focus on materials science applications, where interpretable predictive modeling may provide principled understanding of materials systems and guide new materials discovery. It is often computationally prohibitive to exhaust all the potential relationships to construct and search the whole feature space to identify interpretable and predictive features. We develop and evaluate new AFE strategies by exploring a feature generation tree (FGT) with deep Q-network (DQN) for scalable and efficient exploration policies. The developed DQN-based AFE strategies are benchmarked with the existing AFE methods on several materials science datasets.

Introduction

Feature engineering (FE) is to create features that capture hidden dependency relationships in data and improve the prediction performances of machine learning (ML) algorithms (Brownlee 2014). It usually involves two steps: generating new feature representations by transforming the original raw or primary features in the given dataset, and selecting those engineered features that are important (interpretable and predictive) for the ML tasks. Such prepossessing pipelines and data transformations are crucial and often take most of the actual efforts in deploying ML algorithms (Bengio, Courville, and Vincent 2013; Heaton 2016).

However, traditional FE is a labor-intensive and time-consuming task, which requires complex exercises being performed in an iterative manner with trial and error, driven by domain knowledge developed over time (Bengio, Courville, and Vincent 2013; Khurana et al. 2016). Thus such methods are usually problem-specific and not generally applicable to different datasets, limiting their direct adoption in corresponding applications, especially when both domain knowledge and available training data are scarce. Compared to traditional FE methods, Automated Feature Engineering (AFE) (Khurana et al. 2016; Kaul, Maheshwary, and Pudi 2017) has attracted much attention in recent literature. For example, many black-box deep neural network based AFE models (Long, Lu, and Cui 2019; Fan et al. 2019) have shown their great potential to improve the corresponding ML algorithms' performance and be general to be implemented on different datasets without too much additional manual labor. However, on one hand, the brute force way to generate and select features by exhausting the possible feature transformations takes too much time and is difficult to scale up with the number of original raw features. On the other hand, a good interpretability to the generated features is usually hard to attain through such black-box methods.

For the materials science problems we study in this paper, finding the actuating mechanisms of a certain property or function and describing it in terms of a set of physically meaningful variables is the desired scientific solution (Ghiringhelli et al. 2015). Such a set of physical variables with corresponding parameters that uniquely describe the material and its function of interest, can be denoted as "descriptors". One of the purposes of discovering descriptors in materials-science data, is to predict a target functional property of interest for a given complete class of materials (Ghiringhelli et al. 2017). Hence, AFE for materials science faces two main challenges to build better ML models: a good interpretability of the engineered descriptors from raw features, and the scalability and efficiency to select important descriptors from the often enormous generated feature space for the given target of interest.

A fundamental paradigm in materials science is the existence of causal relationships–typically referred to as Process-Structure-Property (PSP) relationships–that connect processing (i.e., the modifications to a material current state), structure (i.e., the multi-scale arrangement of the material), and properties (i.e., the response of the material to an external stimulus). The navigation of this PSP space is enormously resource-intensive, regardless of whether this query is on physical experiments or computational ones. As a result, it often takes more than two decades to identify, develop, and finally deploy a novel material in real-world applications—a key bottleneck that Materials Genome Initiative (MGI) tried to resolve. Attempting to use physics-agnostic machine learning methods to infer these relation-

ships is limited by the scarcity of the available training data. Moreover, one would be interested in discovering relationships that connect features to properties/behaviors as these relationships can be further exploited to design/discover materials with optimal properties. AFE enables us to use physics constraints in the machine learning models and also enables the discovery of design rules for materials based on fundamental principles.

In this paper, we propose a Feature Generation Tree (FGT) and focus on novel AFE strategies by combining FGT exploration with Deep Reinforcement Learning (DRL) (Mnih et al. 2015) to address both the interpretability and scalability challenges. Instead of employing a brute-force way to perform algebraic operations on the raw features in a given dataset and then selecting important descriptors, we combine the generating and selecting processes together by constructing FGTs and developing the corresponding tree exploration policies guided by deep Q-Network (DQN). An efficient exploration of the prominent descriptors can be attained in the growing feature space based on the allowed algebraic operations, and our new AFE strategies, constructing interpretable descriptors based on a list of operations according to the DRL learned policies, are more scalable and flexible with the performance-complexity trade-off with the help of adjustable batch size for generating intermediate features. More critical to materials science and other scientific ML (SciML) problems, our FGT provides a flexible framework for incorporating prior knowledge (e.g. physics constraints) to generate and select features. This is important for interpretable learning with physics constraints under data scarcity and uncertainty as the space connecting intrinsic materials attributes/features to materials behavior is vast, sparse, and complex in nature.

We have benchmarked different feature exploration strategies for FGT, such as one-step greedy and Monte-Carlo tree search. Deep Q-network (DQN)-guided FGT has shown better empirical performance, by which we achieve better trade-offs between efficiency, scalability, and accuracy. Compared to the exhaustive method in (Ouyang et al. 2018) that screens all features given the complexity, our AFE is more scalable and efficient. Furthermore, compared to other AFE methods based on DQN and graph search, our AFE can identify complex non-linear features while abiding by physics principles, which leads to interpretable features of functional forms with good predictive power by adopting a linear regressor or classifier on top of them.

Related Work

Desirable FE should attain considerable improvement of prediction performance, generalizability, as well as good interpretability with little manual labor. Thus, Deep Feature Synthesis (Kanter and Veeramachaneni 2015), extracts features based on explicit functional relationships without experts' domain knowledge through stacking multiple primary features and implementing operations or transformations on them. But it suffers from efficiency and scalability problems due to its brute-force way to generate and select features. Kaul et al. (Kaul, Maheshwary, and Pudi 2017) proposed Autolearn by regression-based feature learning through mining pairwise feature associations. While it avoids overfitting, to which deep Learning based FE methods are amenable, and improves the efficiency by selecting subsets of engineered features according to stability and information gain, it does not directly produce intepretable features. Khurana et al. (Khurana et al. 2016) introduced Cognito, which formulates the feature engineering problem as a search on the transformation tree with an incremental search strategy to explore the prominent features and later extended the framework by combining RL with a linear functional approximation (Khurana, Samulowitz, and Turaga 2018) to improve the efficiency. A similar framework has recently been developed by Zhang et al. (Zhang et al. 2019), who also used a tree-like transformation graph with the DRL policy. It improves the policy learning capability compared to Cognito. However, both frameworks do not explicitly incorporate available prior knowledge into the AFE procedures.

For AFE in materials science applications, several methods have been developed, such as the method based on compressed sensing (Ghiringhelli et al. 2017) and more recent Sure Independent Screening and Sparse Operation (SISSO) (Ouyang et al. 2018) by brute-force search to generate and select subsets of generated features by the sure independent screening (Fan and Lv 2008) together with sparse operators such as Least Absolute Shrinkage and Selection Operator (LASSO) (Tibshirani 1996). These methods pose an scalability challenge with the exponentially growing memory requirement to store intermediate features and high computational complexity to search for features.

Methodology

In this section, we introduce our new AFE strategies, which are based on the formulated feature generation tree (FGT) exploration guided by deep Q-network (DQN). More critically, we facilitate a flexible intermediate feature generation procedure that helps achieve good performance-complexity trade-off as well as flexible integration of available physics constraints as prior knowledge.

Engineering Descriptors in Functional Forms

Given a dataset $D_0 =< F_0, y >$, where F_0 denotes the finite set of p variables as raw or primary features $\{f_0^0, f_0^1, ..., f_0^p\}$ and y denotes the target vector, we need to construct sets of engineered features $F_i = \{g_1(F_0, c_1), g_2(F_0, c_2), \dots\}$ based on functional forms with allowed algebraic operations to generate interpretable and predictive descriptors for y. The function $g_m(\cdot)$ consists of a set of algebraic operations ϕ, from an operation set O, implemented on features in F_0. The operation set O can be pre-defined, for example, with the following unary and binary operations:

$$O = \{exp(\cdot), log(\cdot), (\cdot)^2, (\cdot)^3, (\cdot)^{-1}, \sqrt{\cdot}, \sqrt[3]{\cdot}, +, -, \times, \div\}. \quad (1)$$

For each function, c_i denotes the complexity of the corresponding generated feature—the number of algebraic operations. For example, the function $\exp(f_0^0) \times (f_0^1)^2 + \sqrt{(f_0^2)}$ has the complexity of 5. Here, F_i denotes the iteratively generated feature set with the maximum allowed complexity c_i.

From the whole feature space $\mathbb{F} = F_0 \cup F_i's$, our goal is to find such an optimal feature set $F^* = \{(f^1)^*, (f^2)^*, \dots\}(\forall (f^d)^* \in F^*, (f^d)^* \in \mathbb{F})$ that maximizes the prediction performance score, for example by classification or regression accuracy, $A_L\{F', y\}$:

$$F^* = \underset{\forall f^k \in F', f^k \in \mathbb{F}, c_i < c_{max}}{\arg\max} A_L\{F', y\}, \quad (2)$$

where L denotes the prediction model, which can be linear regression or Support Vector Machine (SVM) for interpretability with generated features, and F' denotes the set of all generated features.

Feature Generation Tree (FGT)

To approximate the optimal feature set F^*, we introduce the Feature Generation Tree (FGT) illustrated in Figure 1 to iteratively construct the feature space and transform the problem into a tree search problem for efficient AFE. Each node in FGT represents a feature set F_i and each edge represents an operation ϕ. We denote $(F^d)^* = \{(f^1)^*, (f^2)^*, \dots, (f^d)^*\}$ as the top d optimal features when we choose the cardinality of F^* as d, and $(f^d)^*$ as the selected optimal feature for the dth dimension of $(F^d)^*$. The FGT exploration aims to search for the optimal features $(f^1)^*, (f^2)^*, \dots$ one by one. The corresponding complete AFE procedure constructs the feature subspace F^d sequentially as the search space of each $(f^d)^*$ exploration. Iterations start from the root node F_0, which represents the primary feature set. At some node F_i, we choose an operation ϕ_i according to the policy π as detailed in the following subsection, then move forward to the next node, generate the new feature set $F_j (j > i)$ and add the generated features to F^d. Meanwhile, the generated new feature set $F' = (F^{d-1})^* \cup \{f^d\}$ will be fed to the predictive model L to obtain the score $A_L\{F', y\}$, where $(F^{d-1})^* = \{(f^1)^*, (f^2)^*, \dots, (f^{d-1})^*\}$ for each $(f^k)^* \in F^k (1 \leqslant k \leqslant d-1)$, representing the top $d-1$ optimal feature set chosen from the previous feature subspace F^k; and $f^d \in F^d$. The FGT will grow by repeating the operations above until it attains the maximum complexity c_{max}. Then a new iteration will start again to grow FGT to find $(f^d)^*$. When F' achieves our desired prediction performance score, we will stop or look for the next optimal descriptor $(f^{d+1})^*$. Note that the feature subspace F^d is the union of all F_i's in all explored FGT's when looking for $(f^d)^*$, and the whole feature space $\mathbb{F}$ is the union of all F^d's.

Reinforcement Learning for FGT Exploration

AFE by FGT exploration can be considered as a finite Markov Decision Process (MDP) problem. Considering the huge feature space and the large available operation set we may have, we adopt DQN (Mnih et al. 2015) with experience replay to learn the policy π for choosing ϕ's during FGT exploration. Formally, we define the states, actions and rewards for our AFE formulation as follows:

- **state** F_i^d, denoting a primary or generated feature set when looking for the dth optimal descriptor;
- **action** $\pi(F_i^d) = \phi_i$, denoting an operation in the set **O**;

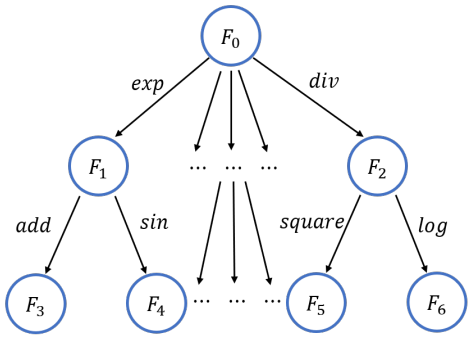

Figure 1: Example of a feature generation tree

- **reward**: $R(F_i^d, \phi_i) = \max_{F'}(1.001 - A_L\{F', y\})^{-1}$, where $0 \leqslant A_L\{F', y\} \leqslant 1$ and $F' = (F^{d-1})^* \cup \{f_d\}$, for $(F^{d-1})^* = \{(f^1)^*, (f^2)^*, \dots, (f^{d-1})^*\}$ for each $(f^k)^* \in F^k (1 \leqslant k \leqslant d-1)$ and $f^d \in \phi_i(F_i^d)$.

Each node in FGT represents a state and each edge an action. To find the optimal F^* with the lowest cardinality to meet the desired prediction performance score threshold, our AFE will first look for $(F^1)^*$, and then look for $(F^2)^*$, or more if the set of resulting descriptors does not meet the score threshold within the allowed budget (maximum number of exploration iterations). To have a flexible exploration procedure for both performance-complexity trade-off and incorporation of prior knowledge, each $(f^d)^*$ in F^* can be chosen from the top n features with highest rewards in the corresponding feature subspace F^d, composing a candidate set S^d. So $(F^{d-1})^*$ can have multiple combinations according to the whole candidate sets $\mathbb{S} = \{S^1, \dots, S^{d-1}\}$, and F' also has multiple combinations according to different $(F^{d-1})^*$ and f^d. Consequently the reward is computed as the maximum reward over F'.

Algorithm 1 DQN for Automatic Feature Engineering

input: Primary features F_0, Action set O
for $d = 1, 2, \dots$ **do**
  Construct new DQN
  Clear Buffer
  for $episode = 1, 2, \dots, N$ **do**
    for $i = 0, 1, \dots,$ **do**
      $\phi_i = \epsilon$-Greedy Method(F_i, ϵ)
      $F_{i+1}^d, R_i, c_{i+1} =$ FGT_Grow $\{F_i^d, \phi_i, c_i\}$
      Buffer $\leftarrow \{F_i^d, F_{i+1}^d, \phi_i, R_i, c_{i+1}\}$
      Train DQN with experience replay
      if $R_i > threshold$ **then**
        goto Output
      end if
      if $c_i \geq c_{max}$ **then**
        break
      end if
    end for
  end for
  $\mathbb{S} \leftarrow$ Candidate set S^d with n features of highest R_i
end for
Output: Optimal feature set F^* chosen from $\mathbb{S}$

Algorithm 2 FGT_Grow

input: Feature set F_i^d, action ϕ_i, complexity c_i
if ϕ_i is unary **then**
    $F_{i+1}^d = \phi_i(F_i^d)$, $c_{i+1} = c_i + 1$, $R_i = R(F_i^d, \phi_i)$
else
    $L = \emptyset$
    Randomize B from feature space $\mathbb{F}$
    for each f^s **in** B **do**
        $L \leftarrow \phi_i(F_i^d, f^s)$
    end for
    Select F_{i+1}^d from L with maximum $R(F_i^d, \phi_i)$
    $R_i = R(F_i^d, \phi_i)$, $c_{i+1} = c_i + (c_s \text{ of } f^s) + 1$
end if
$\mathbb{F} \leftarrow F_{i+1}^d$
Output: Feature set F_{i+1}^d, Reward R_i, Complexity c_{i+1}

It is worth noticing that we have different types of algebraic operations in O. When we apply unary operations ϕ_u on a feature set F_i, it will apply ϕ_u on all the features in F_i and results in the new generated feature set $F_j = \{\phi_u(f^1), \phi_u(f^2), \dots\}$. However, when we apply binary operations ϕ_b on F_i, beside the one feature in the F_i, we have to choose another one feature in the whole feature space to complete the operation, resulting the exploding of the corresponding feature subspace with the new generated feature set as $F_j = \{\phi_b(f^1, f^s)\} \cup \{\phi_b(f^2, f^s)\} \cup \dots$. Clearly, if we enumerate f^s from all the features in $\mathbb{F}$, it is computationally prohibitive as F_j grows exponentially. Thus, we introduce the flexible batch sampling to randomly sample a feature subspace B from $\mathbb{F}$ as a "Batch Set" each time and enumerate f^s only from B to achieve the performance-complexity trade-off, and take the maximum reward from all the combinations as the reward. When prior knowledge is available as physics constraints on applying corresponding operations to specific feature groups, this batch sampling procedure can naturally take care of them.

The pseudo-code for the basic AFE strategy of FGT exploration with DQN is provided in Algorithms 1 and 2. Let us denote the number of raw features, unary and binary operations, desired descriptors, and the maximum number of allowed operations to one feature by P, N_u, N_b, d, and r. Further we denote the batch size in AFE by B, and the candidate set in AFE and subspace in SISSO by S. The computational complexity of our AFE by DQN-guided FGT exploration is then given by $O(S^{d-1}P\sum_{i=0}^{r} N_u^{r-i}N_b^i B^i)$. By contrast, SISSO has a complexity of $O(\sum_{i=0}^{2^{r-1}} P^{2^{r-i}} N_b^{2^r-i-1} N_u^i) + O(S^d)$, increasing double exponentially with r.

Experiments

To evaluate our proposed AFE strategies, we perform experiments with three real-world materials science datasets: one for classification of metal/non-metal materials, one for regression to get alloy elastic behavior based on alloy compositions, and the third dataset for predicting material's phase transition temperature with the physics constraints of feature groups. In these experiments, we assume that we have a limited computation budget and set the same upper bound of the allowed runtime for each experiment. We run all the experiments on the platform with the hardware configuration of Intel Xeon E5-2670, 64GB 1866MHz RAM and 2 NVIDIA k20 GPUs. In the following experiments, "descriptors" denote the engineered features in the final optimal feature set. For each dataset, we perform each algorithm with the same setup 10 times to report descriptors based on the best performing sets of descriptors obtained in the given runtime budget in one run, as well as use the average of the size of feature space and running time for the scalability and efficiency comparison.

For DQN, we have adopted a two-layer Q-network with the corresponding hidden dimensions {150,120} and the $relu$ activation function is used for both layers. The following hyperparameters are set for DQN training: Learning rate: 0.001; Experience replay batch size: 64; Gamma: 0.99; Epsilon: 1.0 (decay 0.99 and min 0.05). Our code is open-source and available at https://github.com/ziyux/AFE.

Classification

The classification problem is based on a dataset of 10 prototype structures $(NaCl, CsCl, ZnS, CaF_2, Cr_3Si, SiC, TiO_2, ZnO, FeAs, NiAs)$ with a total number of 260 materials from one of the experiments reported in Ouyang et al. (2018), which includes seven primary features $(IE_A, IE_B, \chi_A, \chi_B, x_A, x_B, V_{Cell}/\sum V_{atom})$ for each material. The classification problem is to predict whether a given material is metal or not.

In this set of experiments, we test three different kernels as the classification model, including linear Support Vector Machine (SVM), Logistic Regression (LR), and 5-Nearest Neighbours (5-NN). We generate descriptors based on an operation set $\mathbf{O} = \{exp(\cdot), log(\cdot), (\cdot)^2, \sqrt{\cdot}, +, -, \times, \div\}$ to search for a set of two descriptors. The upper bound of the runtime for each descriptor is set to 8 hours and the maximum feature complexity c_{max} is 7 for fair comparison with SISSO in Ouyang et al. (2018).

When applying binary operations to generate new features, we explore different sizes of Batch Sets, denoted as B in Table 3, to evaluate the performance-complexity trade-off by limiting the total number of combinations of feature pairs for corresponding operations to speed up the feature generating process. To study the influence of the size of Batch Sets, we have tested five batch sizes: $\{1,000, 2,500, 5,000, 7,500, 10,000\}$. The generalizability of generated descriptors is evaluated by hold-out testing, first randomly splitting the dataset with 182 materials in the training set and the remaining 78 materials in the test set (7:3). As materials belonging to the same prototype may share information, to further guarantee that the AFE and model training processes do not get additional information about testing data, we also adopt another hold-out testing setup by taking one prototype of materials, CaF_2, to be the test set, and all the remaining prototypes for training. In this setup, 225 materials are for training set and 35 for testing.

In Table 1, we compare our AFE strategies with primary features, One-step Greedy FGT exploration, and SISSO (Ouyang et al. 2018). With random hold-out test,

Kernels	Methods	Descriptors	Accuracy	
			Training	Prediction
Linear SVM	Primary Features	$IE_A, IE_B, \chi_A, \chi_B, x_A, x_B, V_{Cell}/\sum V_{atom}$	0.9780	0.9615
	SISSO	$(f^1)^* = \frac{IE_A IE_B}{\chi_A x_B}(\chi_A^2 - \chi_B \frac{V_{cell}}{\sum V_{atom}})$ $(f^2)^* = IE_B^2(x_A + \frac{V_{cell}}{\sum V_{atom}})(\frac{V_{cell}}{\sum V_{atom}}/\chi_A - \frac{\chi_A}{\chi_B})$	0.9890	0.9487
	One-step Greedy guided FGT	$(f^1)^* = \chi_A - \chi_B + \log \chi_A - \frac{V_{Cell}}{\sum V_{atom}}$ $(f^2)^* = \chi_A^2 - \chi_A \chi_B - \frac{2x_A}{\exp \chi_B}$	0.9615	0.9743
	DQN-guided FGT	$(f^1)^* = \sqrt{\exp \chi_A} - \frac{V_{Cell}}{\sum V_{atom}}$ $(f^2)^* = \log x_A + (\chi_A * IE_A)\frac{V_{Cell}}{\sum V_{atom}}$	0.9890	0.9872
Logistic Regression	Primary Features	$IE_A, IE_B, \chi_A, \chi_B, x_A, x_B, V_{Cell}/\sum V_{atom}$	0.9615	0.9615
	SISSO	$\frac{IE_A IE_B}{\chi_A x_B}(\chi_A^2 - \chi_B \frac{V_{cell}}{\sum V_{atom}})$ $(f^2)^* = IE_B^2(x_A + \frac{V_{cell}}{\sum V_{atom}})(\frac{V_{cell}}{\sum V_{atom}}/\chi_A - \frac{\chi_A}{\chi_B})$	1.0000	0.9487
	DQN-guided FGT	$(f^1)^* = 2 * log\chi_A - V_{cell}/\sum V_{atom} - \chi_B$ $(f^2)^* = \chi_B * log\chi_A/IE_A$	0.9838	0.9743
5-NN	Primary Features	$IE_A, IE_B, \chi_A, \chi_B, x_A, x_B, V_{Cell}/\sum V_{atom}$	0.9450	0.8846
	SISSO	$\frac{IE_A IE_B}{\chi_A x_B}(\chi_A^2 - \chi_B \frac{V_{cell}}{\sum V_{atom}})$ $(f^2)^* = IE_B^2(x_A + \frac{V_{cell}}{\sum V_{atom}})(\frac{V_{cell}}{\sum V_{atom}}/\chi_A - \frac{\chi_A}{\chi_B})$	1.0000	0.9487
	DQN-guided FGT	$(f^1)^* = (IE_A + \chi_B) * IE_B * \chi_B/(exp\chi_A - (\frac{V_{cell}}{\sum V_{atom}})^2)$ $(f^2)^* = log(IE_B/V_{cell}/\sum V_{atom}) - x_A$	1.0000	0.9487
Linear SVM with Type Split	Primary Features	$IE_A, IE_B, \chi_A, \chi_B, x_A, x_B, V_{Cell}/\sum V_{atom}$	0.9822	0.9428
	SISSO	$(f^1)^* = \frac{IE_B(\chi_A + IE_A)}{\chi_A} \exp(\frac{V_{cell}}{\sum V_{atom}}/\chi_A)$ $(f^2)^* = (\frac{IE_B}{\chi_A} + \frac{V_{cell}}{\sum V_{atom}}/x_B)(IE_B + \frac{V_{cell}}{\sum V_{atom}})\frac{V_{cell}}{\chi_A \sum V_{atom}}$	1.0000	0.6000
	DQN-guided FGT	$(f^1)^* = \log \chi_B - (\chi_A/\frac{V_{Cell}}{\sum V_{atom}})^2$ $(f^2)^* = x_B + \log IE_A + \sqrt{IE_B + \frac{V_{Cell}}{\sum V_{atom}}}/\chi_A$	0.9689	0.9143

Table 1: Metal vs. non-metal classification descriptors

we can see that with linear SVM, both One-step Greedy and DQN-guided FGT strategies can engineer predictive descriptors with the improved prediction accuracy compared to the model with primary features. When constructing features using training samples, SISSO and DQN-guided FGT strategies can get the best training accuracy. However, as shown in the table, SISSO is prone to overfitting and has a rather lower prediction accuracy than the model with primary features. By contrast, DQN-guided FGT exploration has more stable and significant improvement on the prediction accuracy. For LR, SISSO has the best training accuracy, but again it overfits. Our DQN-guided FGT strategies can provide both improvement on the training and prediction accuracy over primary features, and has the highest prediction accuracy. For 5-NN, both SISSO and DQN-guided FGT achieve the same prediction power, significantly improving the training and prediction accuracy over primary features, but both suffer from overfiting due to the non-linearity nature of the 5-NN classifier. We note that as the engineered descriptors are often non-linear, it is desirable to simply adopt linear predictors based on these descriptors for generalizable and interpretable learning. For hold-out test split by material type, SISSO has the best training accuracy but overfits. Our DQN-guided FGT has better performance than SISSO but is not better than the model with primary features. This demonstrates the difficulty in applying machine learning methods in materials science. Materials systems are often heterogeneous with potential non-linear phase transitions. Without physics knowledge or sufficient data, simpler predictive models may be more robust.

In the rows marked as "classification" in Table 3, we compare the scalability and efficiency of SISSO and DQN by calculating the average number of generated intermediate features and the total runtime to identify interpretable and predictive descriptors. For our DQN-guided FGT exploration, different batch sizes B have been tested and linear SVM is used. In these experiments, all these different setups attain the similar best training accuracy with the same runtime budget. Together with the results in Table 1, we can see that our AFE strategies can attain similar training accuracy and better prediction accuracy compared to the results by SISSO while our strategies construct a much smaller feature space than SISSO's to identify the predictive descriptors. In addition to better scalability, with a suitable Batch Set size, for example, B =1,000 or 2,500, our AFE strategies with DQN can also be computationally more efficient than SISSO, taking less runtime to find the descriptors. With a small batch size, FGT exploration can iterate faster and DQN can also converge to stable policies faster; while with a large batch size, our AFE strategies generate more intermediate features in the binary operation steps and thus can lower the chance of overlooking some promising features with long-term benefits in increasing predictive power.

Kernels	Methods	Descriptors	R2 Score RMSE	
			Training	Prediction
Linear Regression	Primary Features	$C_{11}, C_{12}, C_{44}, K, G, E, AR, CR,$ $ea, phi, m, rho, val, chi$	0.9736 13.923	0.9545 17.045
	SISSO	$(f^1)^*$ = $(C_{11} - K)C_{12}/m - \sqrt{Kea}$ $(f^2)^*$ = $(E - K)^2/(phi/AR - rho/chi)$ $(f^3)^*$ = $val - \log C_{44}/\log chi$	0.9887 9.115	0.9654 15.742
	One-step Greedy guided FGT	$(f^1)^*$ = $val - CR\sqrt{C_{44}}/val$ $(f^2)^*$ = $val + m/(AR \cdot C_{12})$ $(f^3)^*$ = $\sqrt{rho} + \frac{\exp CR}{AR^2}$	0.9804 12.013	0.9647 15.906
	DQN-guided FGT	$(f^1)^*$ = $val - (CR - ea)/\log E$ $(f^2)^*$ = $(\exp AR \log E)/\log rho$ $(f^3)^*$ = $(\exp CR + chi - val)\exp\sqrt{K}$	0.9813 11.720	0.9708 14.460

Table 2: Alloy elastic behavior regression descriptors with prediction performances

Dataset	Average Performance	Methods					
		SISSO	DQN-guided FGT				
			B=1,000	B=2,500	B=5,000	B=7,500	B=10,000
Classification	Feature Space	1,253,648,288	235,864	330,624	452,598	378,985	473,275
	Runtime (hours)	7.8	4.6	6.2	10.3	8.4	11.1
Regression	Feature Space	84,827,777,031	960,730	1,225,284	823,402	972,289	1,130,305
	Runtime (hours)	35.9	7.7	10.0	9.7	8.2	11.4

Table 3: Scalability and efficiency comparison for metal vs. non-metal classification and alloy elastic behavior regression (Note: 'Feature Space' here refers to the total number of generated intermediate features until the final descriptors are found.)

Regression

We implement our AFE strategies for another materials science dataset studying alloy elastic constants, in which 14 primary features are made from elemental properties using the rule-of-mixtures: crystal radius (CR), atomic radius (AR), electron affinity (ea), ionization potential (phi), melting point (m), density (rho), number of valence electrons (val), electronegativity (chi). And the elastic constants: C_{11}, C_{12}, C_{44}, and bulk modulus (K), shear modulus (G), and Young's modulus (E). For brevity, we refer average properties, e.g., $C_{11average}$ as C_{11} throughout the paper (unless specified). The target elastic constant C_{11m} is the first-principle calculated value for an alloy in the quinary alloy system Mo-W-Ta-V-Nb.

Our DQN-guided FGT strategies are implemented to search for a set of three descriptors with the maximum complexity c_{max} set to 7 for this regression problem. Specifically, we use the operation set $\{exp(\cdot), log(\cdot), (\cdot)^2, \sqrt{\cdot}, +, -, \times, \div\}$ and limit the upper bound of the runtime for each descriptor to be 6 hours for FGT methods. Linear regression is adopted to predict C_{11_m} by engineered descriptors. The prediction performance is evaluated based on the R2 score and root-mean-square error (RMSE). For the training vs. test set split, we randomly separate the dataset with the ratio 7:3, resulting in 58 materials in the training set and 25 materials in the test set. Table 2 shows the best performing descriptors in 10 runs for each methods and Table 3 shows the average size of feature space and running time when DQN with different batch size attain the same maximum training R2 score in the time budget for the first time for scalability and efficiency comparison.

From Tables 2 and 3 (regression rows), all the feature engineering strategies can improve the prediction performance compared with the model with primary features. SISSO has the highest training R2 score with the much higher cost of significantly larger feature space and longer runtime to search for three final descriptors than our AFE strategies with DQN. On the other hand, without considering potential long-term benefits when constructing intermediate features, one-step greedy cannot perform better than SISSO or DQN-based strategies. With small datasets, SISSO again shows the tendency of overfitting. Our AFE strategies with DQN attain similar training R2 scores as SISSO does but have the highest prediction R2 score with the derived descriptors, showing its potential to derive physics meaningful descriptors with better scalability and computational efficiency.

Regression with Physics Constraints

We also implement our AFE algorithm on a physics-constrained dataset, which consists of 14 elemental properties as primary features $\{G_1, G_2, \ldots, G_{14}\}$ for each of 40 possible constituting elements. Here $G_i = \{f_i^1, f_i^2, ..., f_i^{40}\}$ with f_i^j representing the ith property of the jth element. The elemental properties are weighted based on each element's contribution for a given material. We apply our AFE strategies to derive sets of descriptors based on these 14×40 primary features and fit a regression model to predict the target of the transformation temperature A_f.

Kernels	Methods	Descriptors	R2 Score RMSE	
			Training	Prediction
Linear Regression	Primary Features	$G_1, \ldots, G_{14}$	0.7070 78.303	0.5762 97.446
	DQN-guided FGT (Constrained method 1)	$(f^1)^* = (G_3(G_1 + G_5)/G_1)/G_{13}$ $(f^2)^* = (G_7^{\frac{1}{2}} + (G_2(G_1 - G_2)))(G_5 + G_3^{-2})$	0.7301 75.159	0.5816 96.817
	DQN-guided FGT (Constrained method 2)	$(f^1)^* = G_3^2 + (G_1 + G_{10})(\sqrt{G_5} - G_{12})$ $(f^2)^* = G_8(G_2 - G_6 G_{13}) - G_6^2 G_{13} - G_4^2 - G_1^2$	0.7268 75.605	0.5472 100.724

Table 4: Descriptors derived by physics-constrained AFE and their performance (Note: Descriptors for constrained method 1 and 2 can be interpreted as $g_m(G_1, \ldots, G_{14}) = g_m(\sum_j (f_1^j), \ldots, \sum_j (f_{14}^j))$ and $g_m(G_1, \ldots, G_{14}) = \sum_j g_m(f_1^j, \ldots, f_{14}^j)$.)

The reason for the constraints is that, in materials systems, rules must be followed to ensure fundamental science laws not violated. A material can not have elemental contributions that sum to less than or greater than one. To reveal underlying physical meaning, the elemental properties can not be combined in a way that results in an element preference, such as leaving specific elements and the corresponding properties out of the model. These rules constrain our possible AFE into two ways. The first is to sum up all the features with the same property across the different elements, and then apply AFE on them. The second is to implement AFE on features with the same element across different properties and then add them up to perform the regression.

In both implementation methods, we choose the action space to be $\{exp(\cdot), log(\cdot), (\cdot)^2, (\cdot)^{-1}, \sqrt{\cdot}, +, -, \times, \div\}$. Due to the increasing number of primary features in this dataset compared to previous datasets, SISSO is not computationally feasible to identify predictive descriptors within the runtime budget. We report the results based on our AFE strategies with DQN. Specifically, we search for a set of two descriptors of the maximum allowed complexity c_{max} at 15 with the runtime budget of 6 hours for each descriptor. We compute the R2 score and RMSE with linear regression to evaluate the prediction performances of engineered descriptors. For the training vs. test set split, we randomly separate the dataset with the ratio 8:2, resulting in 472 materials in the train set and 118 materials in the test set.

Table 4 shows the results from the two physics-constrained methods. We can see that the derived predictors by the first constrained AFE method achieves the best training and prediction R2 scores and RMSEs, demonstrating again that our AFE can help identify interpretable and predictive descriptors. We note that, although the second constrained AFE method can find the descriptors with better training R2 score than the model with only primary features, the testing R2 score is low. One reason is that the second strategy searches for descriptors considering algebraic operations on primary features from 40 possible elements. With the limited number of training samples, this is more prone to overfitting than the first constrained AFE method operating on the 14 summarized primary features (i.e. $\sum_j f_i^j$). We expect that the performance can be improved when more prior physics knowledge and constraints are available to further restrict the feasible feature space.

We have further compared our AFE with a recently developed physics-informed Genetic Programming (GP) method (Hernandez et al. 2019) to arrive at analytical many-body classical inter-atomic potentials. With the same simulated molecular dynamics data and experimental setup in the paper, our AFE has achieved the total system energy prediction with a mean absolute error (MAE) of 119 meV within 12 hours. By contrast, the reported model GP1 by GP in Hernandez et al. (2019) had a prediction MAE of 132 meV after 360 CPU hours on the same training and test sets.

Conclusions

In this paper, we present a physics-constrained AFE framework based on the DQN-guided feature generation tree exploration for predictive modeling in materials science, where interpretability is critical to help subsequent critical decision making. The recently developed SISSO (Ouyang et al. 2018) generates all the complex features and then selects "explanable" ones by sure independent screening, which requires generating an enormous number of complex features (double exponential with respect to the number of allowed operators for feature generation), leading to the exponential memory and computation complexity. On the other hand, one-step greedy AFE may lose promising features due to the non-monotonic relationship between features and prediction accuracy. Our proposed AFE strategies tackle these problems by approximating the expected future reward of feature generation through DQN, and replacing the exhaustive feature generation by DQN-guided FGT exploration considering physics prior knowledge. Consequently, our AFE enhances scalability and computational efficiency without sacrificing prediction performance.

The results of our real-world materials science experiments have demonstrated the potential of our DQN-guided FGT exploration in reducing the runtime and enhancing the scalability for automatic feature engineering. More importantly, the engineered descriptors are interpretable with the corresponding lists of algebraic operations on the original primary features. Our physics-constrained AFE can be generalized to other sciML problems involving complex systems, where training data tend to be scarce and noisy as obtaining data can be difficult, time-consuming, and costly. Generalizable learning under data scarcity and uncertainty is critical in these problems, and interpretable instead of "blackbox" learning helps new knowledge discovery and better decision making.

Acknowledgements

The authors acknowledge the support by National Science Foundation (NSF) under the award number OAC-1835690. WT and GV acknowledge the support of NSF through Grant No. DGE-1545403 (Data-enabled Discovery and Design of Energy Materials, D^3EM) and QNRF Grant No. NPRP11S-1203-170056. We also thank Texas A&M High Performance Research Computing for providing computational resources to perform experiments in this work.

References

Bengio, Y.; Courville, A.; and Vincent, P. 2013. Representation learning: A review and new perspectives. *IEEE transactions on pattern analysis and machine intelligence* 35(8): 1798–1828.

Brownlee, J. 2014. Discover feature engineering, how to engineer features and how to get good at it. *Machine Learning Process* .

Fan, C.; Sun, Y.; Zhao, Y.; Song, M.; and Wang, J. 2019. Deep learning-based feature engineering methods for improved building energy prediction. *Applied Energy* 240: 35–45.

Fan, J.; and Lv, J. 2008. Sure independence screening for ultrahigh dimensional feature space. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)* 70(5): 849–911.

Ghiringhelli, L. M.; Vybiral, J.; Ahmetcik, E.; Ouyang, R.; Levchenko, S. V.; Draxl, C.; and Scheffler, M. 2017. Learning physical descriptors for materials science by compressed sensing. *New Journal of Physics* 19(2): 023017.

Ghiringhelli, L. M.; Vybiral, J.; Levchenko, S. V.; Draxl, C.; and Scheffler, M. 2015. Big data of materials science: critical role of the descriptor. *Physical review letters* 114(10): 105503.

Heaton, J. 2016. An empirical analysis of feature engineering for predictive modeling. In *SoutheastCon 2016*, 1–6. IEEE.

Hernandez, A.; Balasubramanian, A.; Yuan, F.; Mason, S. A.; and Mueller, T. 2019. Fast, accurate, and transferable many-body interatomic potentials by symbolic regression. *npj Computational Materials* 5(1): 1–11.

Kanter, J. M.; and Veeramachaneni, K. 2015. Deep feature synthesis: Towards automating data science endeavors. In *2015 IEEE international conference on data science and advanced analytics (DSAA)*, 1–10. IEEE.

Kaul, A.; Maheshwary, S.; and Pudi, V. 2017. Autolearn—Automated feature generation and selection. In *2017 IEEE International Conference on data mining (ICDM)*, 217–226. IEEE.

Khurana, U.; Samulowitz, H.; and Turaga, D. 2018. Feature Engineering for Predictive Modeling Using Reinforcement Learning. In *Proc of AAAI 2018*, 3407–3414.

Khurana, U.; Turaga, D.; Samulowitz, H.; and Parthasrathy, S. 2016. Cognito: Automated feature engineering for supervised learning. In *2016 IEEE 16th International Conference on Data Mining Workshops (ICDMW)*, 1304–1307. IEEE.

Long, W.; Lu, Z.; and Cui, L. 2019. Deep learning-based feature engineering for stock price movement prediction. *Knowledge-Based Systems* 164: 163–173.

Mnih, V.; Kavukcuoglu, K.; Silver, D.; Rusu, A. A.; Veness, J.; Bellemare, M. G.; Graves, A.; Riedmiller, M.; Fidjeland, A. K.; Ostrovski, G.; et al. 2015. Human-level control through deep reinforcement learning. *nature* 518(7540): 529–533.

Ouyang, R.; Curtarolo, S.; Ahmetcik, E.; Scheffler, M.; and Ghiringhelli, L. M. 2018. SISSO: A compressed-sensing method for identifying the best low-dimensional descriptor in an immensity of offered candidates. *Physical Review Materials* 2(8): 083802.

Tibshirani, R. 1996. Regression shrinkage and selection via the lasso. *Journal of the Royal Statistical Society: Series B (Methodological)* 58(1): 267–288.

Zhang, J.; Hao, J.; Fogelman-Soulié, F.; and Wang, Z. 2019. Automatic feature engineering by deep reinforcement learning. In *Proceedings of the 18th International Conference on Autonomous Agents and MultiAgent Systems*, 2312–2314.