Technische Universität Berlin
Institut für Theoretische Physik
Arbeitsgruppe Nichtlineare Optik und Quantenelektronik

Fritz-Haber-Institut
Novel-Materials Discovery (NOMAD) Laboratory

Masterarbeit

Identifying exceptional data points in materials science using machine learning

Milena Oehlers

Matrikelnr.: 360094

10.11.2021

1. Gutachter: Prof. Dr. Andreas Knorr
2. Gutachter: Prof. Dr. Matthias Scheffler
Betreuer: Dr. Luca Ghiringhelli

Die selbständige und eigenhändige Anfertigung versichert an Eides statt
Berlin, den 10.11.2021

. . . .
Milena Oehlers

Deutsche Zusammenfassung

Ziel dieser Arbeit ist, Methoden zu vergleichen, welche vorhandene Strukturen in einem Festkörperdatensatz zu extrahieren, mit deren Hilfe die Vorhersage der Zieleigenschaft verbessert und eine Gruppe repräsentaiver Datenpunkte für ein effizientes Training gefunden werden kann. Diese *Repräsentanten* sollen reichen, um ein Modell ähnlicher Qualität zu erhalten wie eines, das aus Training auf dem ganzen Datensatz beruht. Eine ihrer Varianten, *Prototypen*, sollen zusätzlich die unterliegenden Clusterstruckturen repräsentieren, während die andere Variante, *Archetypen*, Aussagen über die Qualitätsgrenzen zukünftiger unbekannter Materialien erlauben sollen. Wir untersuchen weiterhin, ob *Besondere Datenpunkte* existieren, welche unabdingbar in der Gruppe der *Repräsentanten* sind. Mit dem *Kubische Perowskit* Datensatz, der zuverlässige Datenqualität und eine ausreichende Größe aufweist, und Metriken zur Methodenevaluation, welche auf dem Median von Test-Fehlern basieren und somit die statistische Signifikanz unserer Resultate sichern, nähern wir uns diesem Ziel indem wir die folgenden drei zentralen Fragen beantworten:

Hat ein gegebener Datensatz eine Struktur, oder Untergruppen von Materialien welche durch andere Gesetze beschrieben werden als andere? Zur Beantwortung dieser Frage bewerten wir die Qualität der extrahierten Strukturen mit dem Median Test-Fehler von *mSISSO*, welches verschiedene Fitting-Koeffizienten für jede Materialgruppe erlaubt.

Einige der untersuchten Clustermethoden basieren auf grundlegenden chemischen Erkenntnissen über den Datensatz *Einfacher Kubischer Perowskite* ABO_3, deren Elemente A alle der Gruppe eins bis drei und dem f-Block des Periodensystems angehören, während die Elemente B alle aus Gruppe vier oder höher stammen. Ihre vergleichende Bewertung zeigt, dass Methode *ElB*, welche alle Materialien desselben Elements B demselben Cluster zuordnet, und Methode *GruppeB*, deren Materialien gemäß der Gruppe ihres B Elements im Periodensystem Clustern zugeordnet werden, mit Abstand am erfolgreichsten sind. Verglichen mit einer *sSISSO* Applikation, dessen Koeffizienten global für alle Materialien gefittet werden, reduziert sich der Median Test-Fehler um mehr als 67%. Eine Anpassung dieser Methoden für eine Anwendung auf generische Datensätzen wird präsentiert. Die Qualität ihrer Resultate ist jedoch unbekannt und muss in zukünftigen Arbeiten evaluiert werden.

Andere Methoden extrahieren prototypische Cluster im primry-Feature-Raum (*PrimaryRaum*), welches die Zieleigenschaft entweder ausschliesst oder einschliesst (*ohneZiel* oder *mitZiel*), oder im durch *sSISSO* abgeleiteten Raum (*Abgeleiteter-Raum*). Von diesen erweisen sich *Kmeans*, welches isotrpische Cluster gleicher Varianz aufspürt, welche somit gleichmässige Unterteilungen des Featureraums repräsentieren, und *HDBSCAN*, welches die genannten Einschränkungen nicht teilt und unterliegende Clusterstrukturen somit besser abbilden kann, als erfolgreich im *PrimaryRaum ohneZiel* und im *AbgeleitetenRaum*, in beiden mit Clusteranzahl $c \in [5, 8]$. Ihr Median Test-Fehler reduziert sich um 30 bis 35%, und die Anwendung auf generische Datensätze ist ohne Anpassung möglich.

Alle oben erwähnten erfolgreichen Methoden sind sowohl global als auch lokal zufriedenstellend, und ihre extrahierten Clusterstrukturen gleichen sich teilweise. Es kann gezeigt werden, dass der Grad der Klarheit der Clusterbegrenzung in den

Daten, und damit der Eindeutigkeit der Aufteilung durch den stochastischen Algorithmus *Kmeans*, ein Indikator für die Nützlichkeit seiner Resultate für unsere Zwecke ist.

Wir können somit unsere erste Frage bejahen- die vorgestellten Methoden extrahieren nützliche unterliegende Strukturen. Jede ihrer Materialgruppen gehorcht leicht anderen Gesetzen als die anderen, in dem Sinne dass eigene *mSISSO* Koeffizientenfits für jede von ihnen die Ergebnisse erheblich verbessern.

Aufbauend auf solchen Strukturen wollen wir in einem weiteren Schritt folgendes beantworten:

Kann der Datensatz erheblich reduziert werden, so, dass Training auf ihm weiterhin Vorhersagen von ähnlicher Qualität erlaubt, und existierten stabile oder einzigartige Datenpunkte, deren Inklusion in den so reduzierten Datensatz für den Qualitätserhalt unabdingbar sind?

Für den ersten Teil dieser Frage nutzen wir den Leistungsindikator *Normalisierter Medianfehleranstieg* (*nmei*), welcher möglichst klein sein soll.

Ein globaler Vergleich zeigt, dass die erfolgreichsten Clustermethoden erneut auf grundlegenden chemischen Erkenntnissen, gepaart mit einer *Geführten Zufallsselektion*, beruhen. Methode *ElA*, welche alle Materialien desselben Elements *A* einem Cluster zuordnet, erzielt mit Abstand die besten Resultate, mit einem Median Test-Fehler, der sogar unter dem eines global trainierten *sSISSO* Modells liegt. Die nächsten erfolgreichen Methoden sind *GruppeA*, deren CLuster durch die Gruppe des *A* Elements im Periodensystem bestimmt sind, *ElB*, sowie *TypA*, welches Materialien basierend auf dem Typ von Element *A* entweder zum Alkalimetall-, Alkalisches-Erdmetall-, Lanthadin-, oder Übergangsmetall-Cluster zuordnet. Ihre durchschnittlichen *nmei* bewegen sich zwischen 31 und 66%. Es muss jedoch beachtet werden, dass die Qualität der Ergebnisse der präsentierten Anpassung dieser Methoden für generische Datensätze noch nicht evaluiert wurde.

Weitere erfolgreiche Methoden sind *Clusterzentrumsauswahl* basierend auf *Kmeans* im *AbgeleitetenRaum*, welches marginal bessere Ergebnisse liefert als die Anwendung im *PrimaryRaum ohneZiel*, gefolgt von *Zufallsauswahl* in Clustern, die durch *Kmeans* oder *HDBSCAN* im *PrimaryRaum ohneZiel* und *mitZiel* oder im *AbgeleitetenRaum* extrahiert werden. Ihre *nmei* liegen zwischen 69 und 92%, und vor allem *Zufallsauswahl* bewirkt nur eine marginale Verbesserung der Ergebnisse.

Ein verlässliches Set von optimalen Hyperparametern für diese Methoden kann nicht gefunden werden, da Korrelationen der einzelnen Hyperparameter mit *nmei* inkonsistent sind. Für eine finale vergleichende Analyse können Hyperparameter daher nicht zur Methodenvorselektion verwendet werden.

Stattdessen wird letztere für stark reduzierte Datensätze um 90% durchgeführt, mithilfe eines Methodenleistungsindikators, welcher auf einer Validierungsfehlergrenze basiert. Mit diesem Ansatz sind die besten Methoden gegeben durch *Clusterzentrumsauswahl* für *Kmeans* im *AbgeleitetenRaum*, gefolgt von*Kmeans* im *PrimaryRaum ohneZiel* und *Geführte Zufallsauswahl* basierend auf Clustermethode *TypA*.

Der erste Teil unserer zweiten Frage kann daher ebenfalls bejaht werden- ein reduzierter Datensatz kann für das Training verwendet werden; wird er mit den vorgestellten Methoden extrahiert, werden verlässlich bessere Ergebnisse erzielt als

für eine zufallsbasierte Datensatzreduktion. Soll eine starke Reduktion um 90% erfolgen, sind die Methoden, die sich hinsichtlich des letztgenannten Leistungsindikators hervortun, denen mit besten *nmei* vorzuziehen.

Bezüglich des zweiten Teils der Frage kann gezeigt werden, dass keine stabilen oder einzigartigen *Repräsentanten* existieren, welche Teil des Trainngs sein müssen, um ein Modell hoher Qualität zu erhalten. Für *Prototypen* existieren solche *Besonderen Datenpunkte* also nicht. Dieses Resultat gilt vermutlich generell, also auch für andere Datensätze- jedes Material kann als *Repräsentant* deklariert werden oder auch nicht, solange die *Repräsentanten* richtig kombiniert werden.

Unabhängig davon, welcher Leistungsindikator herangezogen wird- fast keine der besten Clustermethoden für unsere zweite Frage nach einer substantiellen, gleichwertigen Datensatzreduktion entspricht einer der Clustermethoden, die in der ersten Frage hilft, unterliegende Datensatzstrukturen für angepasstes Fitting nutzbar zu machen. Liefert eine Methode ein gutes Ergebnis zu einer der beiden Fragen, lässt sich daher keine Aussage zu ihrer Leistungsfähigkeit bezüglich der anderen ableiten.

Unsere dritte Frage lautet:

Wie können wir abschätzen, ob ein neues Material unbekannter Zieleigenschaft adäquat durch unser vorliegendes Modell vorhergesagt werden kann?

Der Versuch, sie zu beantworten, also *Archetypen* mithilfe des *Iterativen Ansatzes* zu finden, scheitert für seine beiden Versionen *Größter Fehler* und *Lineare Archetypen*. Wahrscheinliche Ursachen für das Scheitern werden erörtert und ein Vorgehen vorgeschlagen, mit dem sie in Zukunft vermieden werden können.

Contents

Glossary

Archetypes See *Representatives.* ix, 39–44, 47

DerivedSpace Space spanned by the derived features of *sSISSO* (usually 3d). Each dimension, corresponding to one derived feature, is "streched" by the respective *sSISSO*-coefficient, such that the change in estimated target property for moving positively along any of the *DerivedSpace* axes is equal to 1. 14, 15, 20, 21, 23–26, 29, 34, 35, 37, 45, 46

DVIB Deep variational information bottleneck. 10

Exceptional Data Points Stable or unique *Representatives* whose inclusion into *Training Set* is strictly necessary in order to obtain a model of similar quality as the one resulting from training on the *Work Set.* 3, 35, 37, 47

GLRM Generalized linear rank model: Principal component analysis with regularization. 1, 2

HDBSCAN Hierarchical density based clustering. vi, 7–9, 14, 15, 17–21, 23–26, 29, 31–35, 45

IQR Interquartile range. 15, 20, 22, 23, 27–29, 33

MaxAE Maximum absolute error. 25, 26

ML Machine learning. vi, 1, 3, 6–8, 10, 12, 43

NN Neural network. 10

NoTarget See *PrimarySpace.* 9, 14, 15, 17–26, 29, 31–33, 35, 45

PCA Principal component analysis. 1

PrimarySpace Space spanned by either primary feature space only, with each feature being standardized (*noTarget*), or primary features and target property, each of which is standardized (*withTarget*). viii, ix, 17–26, 29, 31–35, 45

Prototypes See *Representatives.* 37, 39, 47

Representatives Subset of data set materials which suffice for training *sSISSO* in order to obtain a model of quality comparable to training on *Work Set*. Two versions exist, namely *Average Representatives*, or *Prototypes*, which stand for relevant cluster structures in the data set, and *Extreme Representatives*, or *Archetypes*, which exhibit an additional *Property* that can be used to make limiting statements about the prediction quality of materials of unknown target property. viii, ix, 3, 33–37, 39–43

RMSE Root mean squared error. 15–31, 33, 35–37, 40, 41

(Selection Method Specific) Property If exhibited by *Representatives*, they are considered *Extreme Representatives* or *Archetypes*. Allows to make limiting statementes about the prediction quality of materials of unknown target property- for *Biggest Error Selection Method*, the *Archetypes* are worst described by the respective model as compared to all *Work Set* materials, thus setting an upper limit for prediction errors; for *Linear Archetypes Selection Method*, the convex hull spanned by the *Archetypes* defines a subspace in *sSISSO*'s descriptor space, for which predition quality of materials of unknown target property is probably high. ix, 40, 42–44

SISSO Sure independence screening and sparsifying operator, either single-task (sSISSO), or multi-task (mSISSO). vi, viii, ix, 7, 11–23, 25–33, 37, 39, 40, 43–46

Test Set Fixed subset of data set that is left untouched and only used for final method evaluation. 6, 9, 14, 17–21, 23, 24, 27, 30, 35, 41

Training Set Subset of data set at hand that ML-algorithm is trained on; it is always a subset of the *Work Set*. viii, ix, 6, 13, 15, 27–30, 33, 35–44

VAE Variational autoencoder. 10

Validation Set Subset of data set at hand which remains after subtracting the *Training Set* from the fixed *Work Set*. 6, 35, 39

Volatility Quantified by interquartile range (IQR) or its normalized equivalent (nIQR) and illustrated by boxplots, whose box sizes correspond to IQR, and whose whiskers indicate the range of non-outliers. Whisker positions are given by $[Q1 - 1.5 \cdot IQR, Q3 + 1.5 \cdot IQR]$, with $Q1$ and $Q3$ denoting the first and third quartile. 15, 17, 18, 23, 27–30

WithTarget See *PrimarySpace*. 9, 14, 15, 17–20, 29, 31–33, 35, 45

Work Set Fixed subset of data set at hand that does not belong to the *Test Set*. viii, ix, 6, 9, 14–20, 23, 27, 28, 30–33, 37, 39–44

1 Introduction

The recent surge in applications of machine-learning (ML) algorithms to material science has shown its potential of predicting various properties for the majority of materials inside a given data set. One central aspect of physics however is lost in this approach: Determining the range of validity and thus limitations of the deduced models, which in materials science corresponds to extracting average and extreme representatives.

By combining clustering, variational autoencoders, and supervised ML algorithms, this work aims to find these two types of representatives and explore the following aspects: Does a given data set have a structure, or subsets of materials that follow different laws than others? Can the data set be reduced substantially, such that training the model still yields results of similar quality, and are there stable or unique data points whose inclusion during training is strictly necessary in order to obtain such a model? How can we estimate whether a new material of unknown target property is likely to be predicted well by our current best analytical model? By answering these questions, we intend to pave the way for a ML-driven search for the 'needle in the haystack', with research targeted to promising new materials whose investigated properties differ in the desired way from the rest.

This work is structured as follows: After recapitulating related work on representative data points and defining central terms that are used throughout the thesis, existing ML-algorithm building blocks are presented, whose combinations to *Direct Approach* and *Iterative Approach* are newly introduced in this work to answer our three core questions above. The designs of both approaches are presented subsequently alongside with respective results. A summary recapitulates the core findings and ideas for future research.

1.1 Related work

What exactly are *Exceptional Data Points* and *Representatives* and what do they stand for? In literature, this question has been approached from various slightly different angles and using differing definitions of versions of *Representatives*.

In [1], a general definition via *Principal Component Analysis* (*PCA*) with regularization is proposed. This concept of *Generalized Low Rank Model* (*GLRM*) approximates a data set with n data points and m features as a product of two-dimensional factors by minimizing the objective function

$$\min \sum_{(i,j)\in\Omega} (X_{i,j} - w^i a_j)^2 + \sum_{i=1}^{m} r_i(w^i) + \sum_{j=1}^{n} \tilde{r}_j(a_j) \quad . \tag{1.1}$$

Here, the standardized primary features matrix $X \in \mathbb{R}^{(n\times m)}$ of n data points and m features is to be approximated by a matrix Z of rank $k < m$, which is expressible as

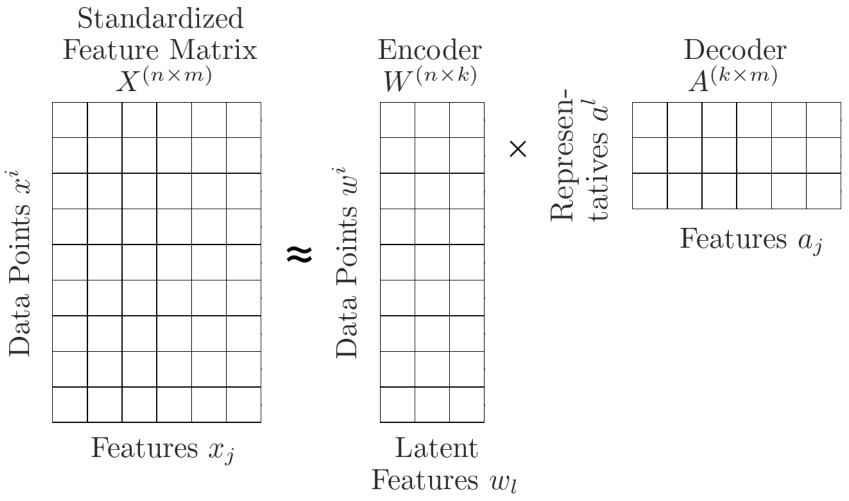


Figure 1.1: Illustration of how to interpret the matrices in *GLRM* [1]. For each data point i, the encoder compresses the m feature values of data point vector x^i to k latent feature values of data point vector w^i. For each feature j, the decoder compresses the n data point values of feature vector x_j to k representative values of feature vector a_j. Each representative vector a^l thus captures the information contained in one of k idealized examples, whose linear combination with w^i maps to data point vector x^i. Figure adapted from [2].

$Z = WA$ with matrices $W \in \mathbb{R}^{(n\times k)}$ and $A \in \mathbb{R}^{(k\times m)}$, while applying the regularizers $r_i, \tilde{r}_j \to \mathbb{R}^\infty$. The ith row of a matrix M is indicated as m^i, the jth column as m_j.

Irrespective of the regularization, this approach can be interpreted in various ways (see fig. 1.1): It can be seen as a bi-linear auto-encoder with encoder W and decoder A, which discovers the latent variables that best explain the observed data X according to squared reconstruction error. As the auto-encoder is low-rank $k < m$, it is an information bottleneck: Each row of encoder W represents a low-dimensional embedding of a data point i by compressing the m feature values of data point vector x^i to k latent feature values of data point vector w^i. The same holds for each column of decoder A, which, for each feature j, compresses the n data point values of feature vector x_j to k representative values of feature vector a_j. At the same time, the rows of A, here denoted *Representatives* a^l, capture the behaviour of k idealized and maximally informative examples. Every data point vector x^i can be expressed as a linear combination w^i of these *Representatives*, where each w^i_l indicates the resemblance of data point i to representative l.

One possible regularization is given by $\tilde{r} = 0$ and non-negativity and a normalization on the rows of matrix W,

$$r = \begin{cases} 0 & \text{if } W_{i,j} \geq 0 \text{ and } ||w_i||_1 \in [0,1] \\ \infty & \text{otherwise} \quad . \end{cases} \tag{1.2}$$

where $||\cdot||_1$ denotes the L1 norm. The constraint $||w_i||_1 \in [0,1] \Rightarrow W_{i,l} < 1$ implies that the position x^i of any data point i that is to be transformed like this is bounded by the *Representatives*' positions a^l for $l \in \{1, ..., k\}$, which are thus *Extreme Representatives* of the data.

Another regularization type is quadratic clustering with $\tilde{r} = 0$ and

$$r = \begin{cases} 0 & \text{if } w^i = e_l \text{ for } l \in \{1, ..., k\} \\ \infty & \text{otherwise} \quad . \end{cases} \tag{1.3}$$

Here, e_l denotes the l-th standard basis vector, such that w^i encodes the cluster to which data vector x^i is assigned. Alternating minimization then yields the cluster centroid position solution $l^* = \arg\min_l \left(\sum_{j=1}^{n} (X_{ij} - A_{lj})^2 \right)$. Quadratic clustering is thus equivalent to *Kmeans* (see section 3.1) and finds *Average Representatives* of the data.

While Udell et al. call both of these specifications *Archetypes* [1], Keller et al. denote only *Extreme Representatives* as *Archetypes*, and *Average Representatives* as *Prototypes* [3] and claim that the assessment of the latter is more useful if a cluster structure exists in the data, and otherwise the former.

Yet another possible interpretation is given in [4], where firstly k *Extreme Representatives*, denoted *Archetypes* in [4], are extracted with the stricter regularization condition $|w_i|_1 = 1$. Data points are allocated in a space spanned by them according to W and cluster memberships are extracted by assigning each data point to the closest of the k *Extreme Representatives.* Corresponding k cluster centroids are determined by maximizing an internal similarity criterion and mapped to the finally selected *Representatives*, denoted *Prototypes* in [4].

1.2 Definitions

The terms *Archetypes* and *Prototypes* are not clear-cut in literature- in this work, we introduce our own definitions of the two concepts in order to be able to answer our three core questions.

In order to assess whether the number of points in a data set can be substantially reduced, we define *Representatives* in general as a subset of materials, upon which training of an ML algorithm yields a model of similar quality as compared to training on the whole data set. If stable or unique *Representatives* can be identified whose inclusion during training is strictly necessary in order to obtain such a model, we call them *Exceptional Data Points.* Depending on the type of *Representatives*, their materials carry additional information: *Average Representatives* or *Prototypes* stand for relevant cluster structures in the data set, for which it is investigated whether some clusters follow different laws than others. *Extreme Representatives* or *Archetypes* on the other hand carry boundary information on the validity of future predictions. A more detailed explanation of the latter is provided in the *Iterative Approach* chapter (5), in which *Archetypes* are mainly assessed, while *Prototypes* are investigated in in the *Direct Approach* chapter (4).

In *ML-Building Blocks* (see chapter 3), we explicitly use the terms archetypes and prototypes in the sense of Keller et al. [3] (see section 1.1), as our definitions do not apply to individual *ML-building blocks.* We additionally stress the difference in definition by not using italics. In *Direct Approach* (see chapter 4) and *Iterative*

Approach (see chapter 5) though, where we present our contributions and investigate our core questions, we refer to our definitions.

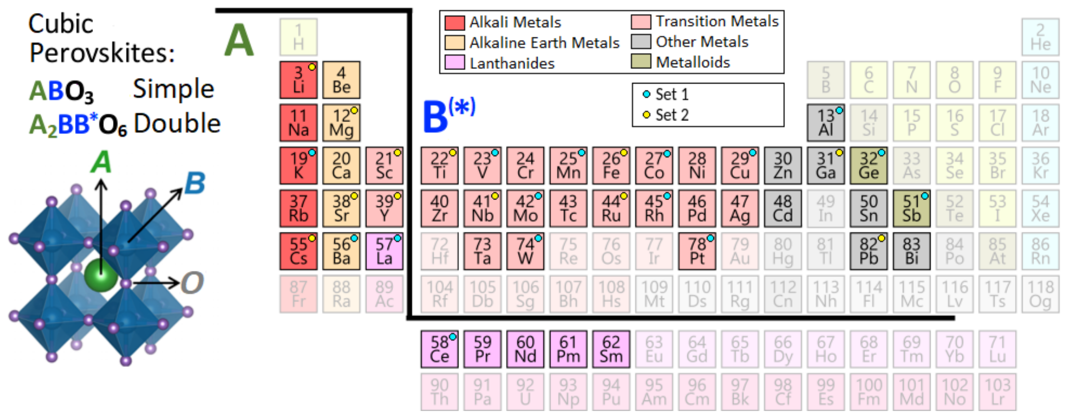


Figure 2.1: Illustration of perovskites atomic structure (left) and their composition (right). All combinations of elements of set A with those of set B constitute the ABO_3 *Simple Cubic Perovskites* data set of 504 materials. In the more general $A_2BB^*O_6$ *Double Cubic Perovskites* data set of 814 materials, all combinations of sets A, B and B^*, where each element belongs to either group one or two as indicated by the upper right circle color, are contained additionally. Figure adapted from Lucas Foppa.

2 Data

In the following, the two versions of the perovskites data set that are investigated in this work are presented, followed by definitions of data set subsets that are used consistently thereafter.

2.1 Two versions of Cubic Perovskites

As this work implements newly combined machine-learning methods, a data set of reliable data quality and sufficient size is needed to be able to assesses the methods' suitability for our goal of finding exceptional data points and determining the exploitability of data set structures.

We thus investigate the ABO_3 *Simple Cubic Perovskites* data set, which is illustrated in fig. 2.1. It comprises all possible combinations of elements in set A, which consists of alkali metals, alkaline earth metals, lanthanides and two transition metals, with the ones in set B, which contains transition metals, other metals and metalloids. In this way, a set of 504 materials is created.

A similar, but larger and more heterogeneous data set version is investigated with the $A_2BB^*O_6$ *Double Cubic Perovskites* of 814 materials. In addition to the 504 *Simple Cubic Perovskites* materials, it contains all combinations of sets A, B and B^*,

Table 2.1: Target property and primary features of the *Simple Cubic Perovskites* and *Double Cubic Perovskites* data set.

Acronym	Meaning	Unit
lat	lattice constant (target property)	Å
r_s, r_p	s/ p orbital radius	Å
Z	atomic number	e
HOMO, LUMO	highest occupied/ lowest unoccupied molecular orbital	eV
EA, IP, EN	electron affinity, ionization potential, electronegativity	eV

where for each material, each constituting element either belongs to set one or two as indicated by the upper right circle color in fig. 2.1, thus yielding 310 additional materials.

2.2 Feature Selection

The investigated target property, the lattice constant, is evaluated with density-functional-theory (DFT) calculations, using the FHI-aims code and the PBEsol exchange-correlation functional. Table 2.1 contains the acronyms and their meanings and units for target property and primary features for both versions of the data set. In formulas, the acronyms are appended a subscript A or B to indicate which element group they refer to.

2.3 Data Subset Definitions

In machine learning literature, terms like *Training Set* are not defined unambiguously. Furthermore, as the way ML algorithms are combined in this work differs from what is typically done, we define the following terms to suit our needs: For any of the above presented data sets, we split data randomly, yielding a *Work Set* and *Test Set*, which stays the same for any methods we apply unless stated otherwise in order to ensure comparability of results. The *Standard Test Set* for *Simple Cubic Perovskites* (ABO_3) is displayed as white fields for the respective combination of material A (vertical axis) and B (horizontal axis) in fig. 4.8. For any application step of the composite algorithm, the *Training Set* is always a subset of or equal to the *Work Set*, whose remaining materials form the *Validation Set*. The *Test Set* is left untouched until final method evaluation, for which the *Test Set* is predicted.

3 ML-Building Blocks

In the following, the algorithms applied to identify archetypes and prototypes in the sense of Keller et al. [3] (see section 1.1) alongside their respective clusters are expounded comparatively. As the terms do not correspond to our definitions introduced in section 1.2, they are not printed in italics.

Afterwards, *Sure Independence Screening and Sparsifying Operator* (*SISSO*) is presented. As opposed to the other building blocks, it extracts analytic equation, which may give physical insight and is thus always used for the final target property prediction and to assess the quality of the extracted data set structure.

3.1 Prototype and Archetype Selection Algorithms

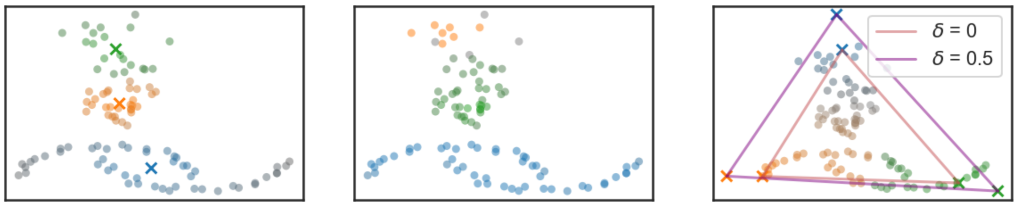
3.1.1 Prototype Identification: Kmeans and HDBSCAN Clustering

Kmeans [5]: This clustering algorithm tries to separate samples into c groups or clusters C of equal variance, minimizing the within-cluster sum-of-squares

$$\sum_{i=0}^{n} \min_{\mu_j \in C} ||x_i - \mu_j||^2 \, , \tag{3.1}$$

where x_i is the position vector of material i and μ_j the one of cluster center j. The number of resulting clusters c is preset- *Kmeans* randomly initializes c centroids, derives the cluster-affiliation of each material with a Voronoi Tesselation [6], thus by assigning it to its closest centroid, and recalculates each cluster's centroid as the average of its members. This is repeated until a (local) minimum is reached. Besides the stochasticity of its result and the need to preset the hyperparameter c, thus the number of clusters, an additional drawback of *Kmeans* is that clusters of unequal variance and density and/or anisotropical distribution are generally not well detected (see fig. 3.1).

All these disadvantages are addressed by the other clustering method, *Hierarchical Density Based Clustering, (HDBSCAN)* [7]. In order to enable finding clusters of varying density, variance and shape and discarding single sparse bridge points between them as outliers, it transforms the space by calculating the mutual reachability distance $d_k(a,b) = \max\{\text{core}_k(a), \text{core}_k(b), d(a,b)\}$ between all material pairs (a,b). Here, $d(a,b)$ denotes the regular Euclidean Distance between a and b, while $\text{core}(a)_k$ calculates the distance of a to its kth nearest neighbor. In a next step, a minimum spanning tree is derived by iteratively adding the edge of lowest mutual distance that connects a lose material to the graph. After sorting the edges by mutual distance, we iterate through them and at each given threshold drop all edges of higher distance. By keeping only the resulting clusters of minimum cluster size m or higher and discarding smaller offsplits as outliers at each threshold,



Kmeans clustering *HDBSCAN* clustering *Linear AA & Voronoi*

Figure 3.1: Prototype / archetype cluster identification as defined by Keller et al. [3] (see section 1.1): For *Kmeans* and *Linear AA* unique virtual representatives can be extracted, which are marked by an **x**. Color saturation indicates how central a point is in its cluster- for *Kmeans* and *Linear AA*, this corresponds to the normed distance of a material to its representative, for *HDBSCAN* to its probability value. *Kmeans* tends to find clusters of equal variance and assign points at the extremes of an anisotropic cluster lower centrality values than *HDBSCAN*. *Linear AA* focuses on extremes instead of averages and hence yields a quite different clustering as obtained by Voronoi Tessalation [6], thus assigning points to their closest archetype. An increased δ and thus higher rate of relaxation of the weighted average constraint leads to a convex hull that encloses more data points.

a dendogram or condensed cluster tree is derived. In order to select those clusters of highest longevity during the distance scan, their respective stability is calculated by $\sum_{p \in \text{cluster}}(\lambda_p - \lambda_{\text{birth}})$, being $\lambda_p = 1/d_k$ the inverse mutual distance of the edge connecting material p to its cluster and λ_{birth} the inverse mutual distance of the edge whose cut leads to the creation of the cluster. Preliminary selecting all leafs and working up the tree, each non-leaf cluster stability is overwritten by the sum of stability of its children, if the latter is larger. If on the other hand the parent stability is larger, the parent cluster is selected and all descendants unselected. When the root cluster is reached, all clusters declared as selected compose the final selection.

As opposed to *Kmeans*, *HDBSCAN* produces outliers which are not assignable to any cluster and its clusters do not have distinguished cluster centers that can be identified as prototypes, which are drawbacks for our purposes.

A *HDBSCAN* version that depends on the number of clusters

All prototype and archetype selection algorithms except *HDBSCAN* depend on the chosen number of clusters c, which are hence scanned as the potential for detection of meaningful clusters might depend highly on it. In order to obtain an equivalent representation for *HDBSCAN*, which selects the number of resulting clusters automatically, we use its deterministic dendogram (see fig. 3.2):

Here, each bar stands for a cluster, whose thickness and color lightness increases with the number of points contained in it. As the inverse mutual distance cutoff λ increases, the bars become thinner, as small clusters containing less points than the

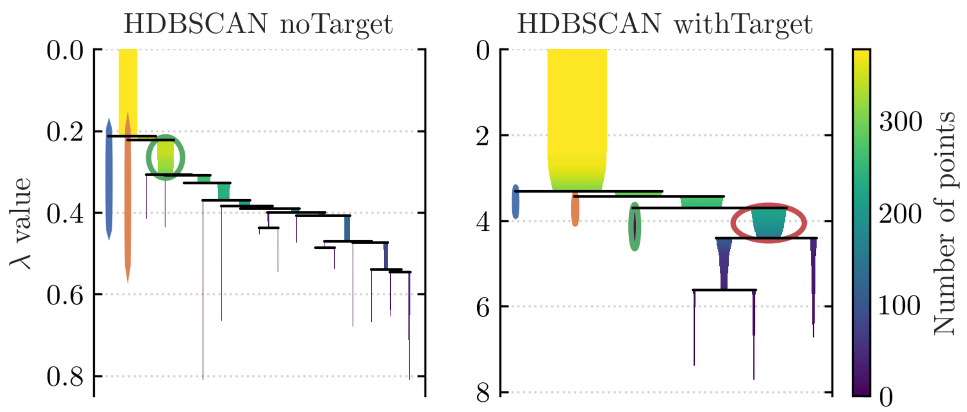


Figure 3.2: Dendogram of deterministic *HDBSCAN*-application for minimum number of clusters $m = 5$, versions *noTarget* (left) and *withTarget* (right), which illustrates the clusters resulting from different inverse mutual distance cutoff values λ. The clusters chosen by *HDBSCAN* are encircled.

minimum cluster size m are split off and discarded as outliers. At the λ where an offsplit cluster size surpasses m, the bar ends and is replaced by two bars indicating the offsplit and remaining cluster.

For *HDBSCAN*-λ we thus scan the dendogram by varying λ, and capture all possible decomposition states as the number of clusters increases. In order to not discard a mayority of points as outliers for higher cluster numbers, the members of each cluster, which appear as λ is increased to its target value, are assigned at the respective cluster split-off λ^{birth}. This process is illustrated in fig. 3.2, right panel: For $\lambda = 4$, our *HDBSCAN*-λ yields the four encircled clusters. If cluster memberships were assigned according to actual decomposition states at $\lambda = 4$, the third cluster from the left would have lost many points as compared to when it first appeared at $\lambda^{\text{birth}} \approx 3.5$, and the first cluster from the left would have dissolved into outliers. For that, cluster member assignments are conducted at their respective birth values, $\lambda^{\text{birth}} \in \{3.1, 3.2, 3.5, 3.5\}$.

The python package *HDBSCAN* does not provide a method to assign unseen test points to the cluster offsplits derived like this. Hence, we introduce an assignment strategy as follows: For each *Test Set* point a, the non-outlier *Work Set* point b of minimal mutual reachability distance $d = \min_b d_k(a, b)$ to it are selected as its potential cluster connection. For any given cluster structure yielded by its associated inverse mutual distance cutoff λ, the latter is then used to determine whether the point at hand is discarded as an outlier for $\lambda > 1/d$, or assigned to the same cluster as its non-outlier *Work Set* point partner for $\lambda < 1/d$.

3.1.2 Archetype Identification: Linear and Deep Archetypal Analysis

Linear archetypal analysis (*Linear AA*) [8, 9] aims to find a preset number A of virtual archetypes of a data set, which span its convex envelope and are calculated as the weighted average of all M data points, $\mathbf{A} = \mathbf{XC}$, where $\mathbf{X}$ is the $(M \times F)$

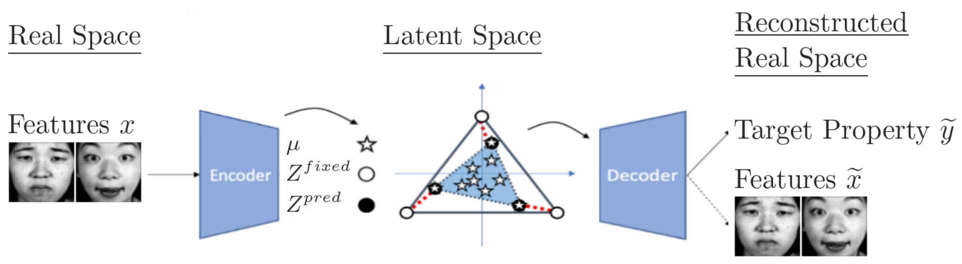


Figure 3.3: *DeepAA* illustration: Its *DVIB* consists of an encoder and two decoder branches of four dense *NN* layers, respectively. The encoder transforms the real space features $\mathbf{X}$ into a latent space representation $\mathbf{T}$ of lower dimension, where the archetypes are extracted, and which is decoded to the real space feature and target property reconstruction, $\tilde{\mathbf{X}}$ and $\tilde{\mathbf{y}}$. Figure adapted from [3].

data point matrix and $\mathbf{C}$ the $(F \times A)$ weighting matrix with $1 - \delta \leq |\mathbf{c}_d|_1 \leq 1 + \delta$ and $\mathbf{C} \geq 0$. The parameter δ allows for a relaxation of the model in case of data points which cannot be enclosed by a convex hull of A virtual archetypes derived like this (which is the typical case for materials data) [9]. The weighted average of these virtual archetypes in turn can be used to approximate the data points themselves with $\tilde{\mathbf{X}} = \mathbf{AS}$ with the $A \times M$ weighting matrix $\mathbf{S}$ with $|\mathbf{s}_n|_1 = 1$ and $\mathbf{S} \geq 0$. Formally, this is achieved by optimizing $\arg\min_{\mathbf{C},\mathbf{S}} ||\mathbf{X} - \tilde{\mathbf{X}}||^2$ (see fig. 3.1).

The two drawbacks of *Linear AA*, namely that it only finds virtual archetypes which have to be mapped to their closest actual data point neighbor, and that its unique optimization solution is not always found by the algorithm due to local minima, is shared by its more sophisticated nonlinear counterpart, *Deep Archetypal Analysis* (*DeepAA*).

Keller et al. [3] introduced *DeepAA* based on a *Deep Variational Information Bottleneck (DVIB)*, a type of *Variational Autoencoder (VAE)*, which was modified for this work for materials data instead of images: In a data set with a $(M \times F)$ data point matrix $\mathbf{X}$, *DVIB* tries to find a latent space representation $\mathbf{T}$ of a preset lower dimension $D < F$ with a dense *Neural Network (NN)* encoder, that contains enough information to reconstruct both the F primary features, $\tilde{\mathbf{X}}$, and a target property, $\tilde{\mathbf{y}}$, with two branches of *NN* decoders (see fig. 3.3). *DeepAA* is thus supervised, contrary to unsupervised *Linear AA*.

As the latent space is artificially constructed by nonlinear transformations and combinations of the real space features, assumptions about its size and shape can and must be made. By definition, the archetypes span the convex hull of latent space of a preset dimension D by being connected through a hyperplane, hence the minimum number of archetypes must be $D + 1$. As the archetypes are to carry unique information, thus, the set of archetypes should be mappable to each data point by an unambiguous weight matrix, the maximum number of archetypes is also $D + 1$. For symmetry reasons, the latent space is always spanned by a simplex of the respective dimension, which is centered in the origin, whose corners are represented by the archetypes position matrix $\mathbf{A}$.

In order to force the encoder to map real space into the so constructed restricted latent space, its intermediate result, a dimension-reduced transformation of real space, is transformed with a final dense layer with softmax activation functions, yielding a weight matrix $\mathbf{S}$ with $|s_n|_1 = 1$ and $\mathbf{S} \geq 0$, whose application to the archetypes position matrix $\mathbf{A}$ produces the position of each data point in latent space. Like this, by construction all data points, including test points that have not been used for network training, are mapped inside the convex hull that is defined by the archetypes.

Mathematically, all primary features $\mathbf{x}$, latent space variables $\mathbf{t}$ and target properties $\mathbf{y}$ are assumed to be normally distributed and the algorithm aims to find a latent space representation that allows for a close feature and target reconstruction, $\tilde{\mathbf{x}}$ and $\tilde{\mathbf{y}}$, by optimizing its objective function

$$\max_{\Phi,\Theta}\{-I_{\Phi}(\mathbf{t};\mathbf{x}) + \lambda I_{\Phi,\Theta}(\mathbf{t};\tilde{\mathbf{y}}) + \nu I_{\Phi,\Theta}(\mathbf{t};\tilde{\mathbf{x}}) - \ell_{\mathrm{AT}}\}\ . \tag{3.2}$$

Here, I denotes the mutual information which is determined by the encoding and decoding functions Φ and Θ, which in turn are determined by their constituting dense layer functions. The function type of each dense layer is preset, such that I is parameterized by the set of assigned weights in all constituting functions. The parameters λ and ν weigh the target and feature reconstruction, respectively, and ℓ_{AT} corresponds to our known *Linear Archetype Analysis* objective function, $\ell_{\mathrm{AT}} = ||\mathbf{Z}^{\mathrm{fixed}} - \mathbf{Z}^{\mathrm{pred}}||^2$, which is here applied to the fixed archetypes $\mathbf{Z}^{\mathrm{fixed}}$, with $\mathbf{Z}^{\mathrm{pred}} = \mathbf{Z}^{\mathrm{fixed}}\mathbf{SC}$.

As opposed to *SISSO* (see next section), a crucial disadvantage of *Deep Archetypal Analysis* is that its target property prediction remains a blackbox as no analytical, human-readable formulas are extracted. Its major advantage over all other prototype and archetype identifying algorithms presented in this work, on the other hand, is that it is supervised- it allows for an archetype definition that includes information about the target property, as the latter is used in the training of the network, while not needing the target property to predict an unseen point's place in latent space. This advantage is bought with high computational cost, though, and a stochastic result.

In order to be able to not only extract representatives but also clusters based on them, after archetype identification we apply a Voronoi Tessalation [6], thus, assign each data point to its closest archetype as illustrated in fig. 3.1 for *Linear AA*. For *DeepAA*, this assignment is based on the distances in latent space. To our knowledge, this combined application has not been used before.

3.2 Sure Independence Screening and Sparsifying Operator (SISSO)

At the core of this work, the supervised compressed-sensing method *Sure Independence Screening and Sparsifying Operator* (*SISSO*) [10] is used to predict a target property vector $\mathbf{p}$ of a compound (for instance its lattice constant) in an analytic, therefore transparent way, which may give physical insight. For this, in a first step it calculates all combinations of its F primary features that are physically possible (thus respecting unit restrictions, e.g. not allowing for summing distances and energies). To this end, it iteratively applies an operator set $\hat{H} = \{I, +, -, *, /, \exp, \log, |-|, \sqrt{}, ^{-1}, ^2, ^3\}$, where I denotes the identity operator, yielding the candidate descriptors matrix $\mathbf{D}$. The maximum number of iterations is given by *rung*, which in this work is always set to $rung = 2$. Typically, $rung = 3$ is necessary for accurate results, but for the method benchmarking conducted in this thesis, conclusions do not depend on the chosen *rung* and choosing $rung = 2$ allows for a more extensive benchmark. In a second step, it determines its best sparse subset, the descriptor or derived features matrix $\mathbf{D}^*$ of dimension $D < F$, for linearly fitting target property vector $\mathbf{p}$, by optimizing $\arg\min_{\mathbf{c}}(||\mathbf{p} - \mathbf{D}\mathbf{c}||_2^2 + \lambda||\mathbf{c}||_0)$, where $||\mathbf{c}||_0$ is the number of nonzero components of the coefficients $\mathbf{c}$.

The derived features space can be seen as a type of latent space as well, which contrary to the *DeepAA* latent space is constrained to be linearly correlated to the target property. Non-linear relationships between primary features and target property are captured in descriptor construction. *SISSO* exhibits two distinguished advantages over *DeepAA* though: Firstly, this deterministic method extracts analytic, therefore transparent formulas, which may give physical insight. Furthermore, *SISSO* does not include the primary feature reconstruction in its objective function, nor other additional terms like the *Archetype Loss* ℓ_{AT}. *SISSO*'s descriptor or latent space is thus constructed by optimizing the reconstruction of the target property only, which leads to a better prediction.

Two different variants of *SISSO* are used: The above-described *Single-task SISSO* (*sSISSO*), which estimates the optimal descriptor and coefficients jointly for all materials, and *Multi-task SISSO* (*mSISSO*) [11]. The latter allows for different coefficients $\mathbf{k}^i$ for given c tasks of M_i materials, respectively, but selects the same descriptor, by optimizing

$$\arg\min_{\mathbf{k}} \sum_{i=1}^{c} \frac{1}{M_i} ||\mathbf{P}^i - \mathbf{D}^i\mathbf{k}^i||_2^2 + \lambda||\mathbf{k}||_0 \; . \tag{3.3}$$

It hence weighs each task by the number of materials inside in order to minimize the global prediction error. How the tasks are defined and thus which materials they contain can be decided for the problem at hand. For instance, clusters as extracted by the algorithms presented in section 3.1.1 can be tasks.

4 Direct Approach

Does a given data set have a structure, or subsets of materials that follow different laws than others (see section 4.1)? Can the data set be reduced substantially, such that training the model still yields results of similar quality, and are there stable or unique data points whose inclusion during training is strictly necessary in order to obtain such a model (see section 4.2)?

In order to answer these questions, in the *Direct Approach*, a data set clustering structure is extracted once with a given clustering method, and either used to assign materials to *mSISSO*-tasks according to their cluster membership in the *Structure Exploitation* section, or to choose *Representative* materials from them with a given selection approach, which then constitute the *sSISSO Training Set*, in the attempt of finding an *Equivalent Reduced Data Set*.

In both the *Structure Exploitation* and *Equivalent Reduced Data Set* section, the same clustering methods are applied in order to be able to compare the results of the two and assess whether a method that is successful with the former approach is so as well with the latter one.

There are three types of clustering methods:

- *Clustering based on element metadata:* Clusters can be constructed based on any meaningful information. These clustering methods each use material metadata of nominal or ordinal scale, which are thus not representable as physical features of the data set as displayed in table 2.1, but which capture central properties of the observed material and might thus be useful for clustering.

  For *ElA/B*, materials of same element A or B, respectively, are assigned to one cluster. As elements A and B determine all primary features, and feature subgroups corresponding to them are completely dependent, this clustering method is more precise and might be more useful than the subsequently introduced *Clustering in Primary Space*. All remaining methods in this clustering type are based on the constituting elements as well and hence share this advantage: For *GroupA/B*, those materials whose element A or B belongs to the same column in the periodic table (see fig. 2.1) are ascribed to the same cluster. For *TypeA/B/AB*, cluster membership is determined by the type as indicated by the coloring of the respective element A, B or both (see fig. 2.1). Hence, for *TypeA* clustering, four clusters are yielded, where element A belongs to the *Alkali Metals*, *Alkaline Earth Metals*, *Transition Metals* or *Lanthanides*, respectively, while for *TypeB* clustering, three clusters are returned containing materials whose element B belongs to *Transition Metals*, *Other Metals*, or *Metalloids*. Consequently their combination, *TypeAB* clustering, results in twelve clusters. In appendix A, the cluster memberships resulting from each clustering method are listed.

The above clustering methods are designed for application to *Simple Cubic Perovskites*- for data sets where a differentiation between elements A, B, etc. is not possible, or where elements exhibit varying stoichiometry, possibly also considering off-stoichiometric compositions, the methods should be adapted. In such a generic data set, any data point i consists of n_i elements e_i^j with composition ratios r_i^j, with arbitrary sequence $j = 1, ..., n_i$ and $\sum_j r_i^j = 1 \; \forall \; i$. In this case, *mSISSO* could be applied to tasks defined by overlapping clustering, which allows each material to simultaneously be part of n_i clusters, which are determined by the affiliation of its elements e_i^j. Each material's fitting coefficients and target property estimation could then be calculated as the weighted mean of its respective *mSISSO* task values, with weights given by the composition ratios r_i^j. Like this, all proposed clustering methods based on *Element*, element *Group* or element *Type* could be applied to generic data sets. The performance of this overlapping clustering approach though is not assessed in this thesis and remains to be investigated in future work.

- *Clustering in Primary Space:* For a given target number of clusters c, the unsupervised clustering methods *Kmeans* or *HDBSCAN-λ* are applied to the space spanned by the primary features as described in table 2.1.

  In general, unsupervised clustering is conducted on the feature space only, without the target property (*noTarget*). In this application version, the distance based clustering algorithms are applied to the primary features, which are each standardized in order to avoid biases. If a structure is yielded that proves to be valuable, it can be used to determine whether an unseen point, whose target property is by definition unknown, is an outlier or if it can be well predicted with the given model. The drawback of this application version is that always the same structure is extracted regardless of the target.

  As intuitively the detection of a more valuable structure is to be expected if information about the target property is included, we additionally apply the *withTarget* version: For that, the same per-feature standardization is conducted as before on all features and the target property. For *Work Set* points, the actual target property is used, for *Test Set* points, whose target property is unknown by definition, its *3d-sSISSO* estimation is used. Then, we multiply the standardized target property by the number of included features, so that its influence on clustering is always as high as the summed up influence of all features and does not depend upon their number, and conduct clustering on the resulting feature-and-target-property space. To our knowledge, this approach has not been used before.

- *Clustering in Derived Space:* For a given target number of clusters c, *Kmeans* or *HDBSCAN-λ* are applied to the three-dimensional *DerivedSpace* as spanned by the derived features d_i for $i \in [1, 2, 3]$ that are yielded by *3d-sSISSO* when applied to the *Work Set*. Each dimension is "stretched" by the respective *sSISSO*-coefficient c_1, c_2 or c_3, such that the change in estimated target property for moving positively along any of the *DerivedSpace* axes is equal to 1. For the case of *DeepAA*, clusters are assigned via Voronoi tesselation in its latent space (see section 3.1).

The *noTarget/ withTarget* distinction does not apply to clustering in derived/ latent space, as the latter is extracted based on both information about primary features and the target property. At the same time, the position of a material in derived/ latent space can be calculated even if its target property is unknown, thus also allowing for outlier analysis.

The performance of a method is measured based on test-root mean squared error (RMSE) of *3d-SISSO* only, as for both *Singletask* and *Multitask*, *3d-SISSO* ususally yields best results on the examined data sets. Similarly, for simplicity the *SISSO* hyperparameter *rung* is set to two, in order to avoid further increasing the complexity of hyperparameter choice. Qualitatively, the obtained results are expected to also hold for all values for *rung*.

All clustering methods based on element metadata and *HDBSCAN-*λ are deterministic, while *Kmeans* and *DeepAA* are stochastic. If the *Training Set* creation algorithm, which consists of the clustering and the subsequently applied selection or assignment method, is deterministic, results are analyzed based on *3d-SISSO* test-RMSE. If it is non-deterministic, corresponding median errors and error volatilities are assessed. The latter is illustrated by boxplots- its box sizes indicate interquartile error ranges (IQR), which are used as a quantification of *error volatility*, while its whiskers indicate the range of non-outliers. Their positions are given by $[Q1 - 1.5 \cdot IQR, Q3 + 1.5 \cdot IQR]$, where $Q1$ and $Q3$ represent the first and third quartile. Both median error and error volatilities are calculated based on 30 independent runs. An exception is given by *Training Set* creation algorithms relying on the computational expensive, non-deterimenistic *DeepAA* clustering method. Its quality in terms of feature and target reconstruction varies considerably, and thus also its quality of corresponding clustering- in order to obtain a suitable clustering but limit computation times, *DeepAA* is run 20 times independently, and the best clustering in terms of feature and target reconstruction is selected. If the following selection or assignment method is stochastic, that step is run 30 times independently, but it is always based on the same best-*DeepAA* clustering.

In section 4.1, we can show that for *Structure Exploitation*, clustering methods *ElB* and *GroupB* are most successful, with a reduction in test-RMSE of more than 67%. For finding an *Equivalent Reduced Data Set* in section 4.2, depending on the success measure, either *ElA* and *GroupA*, or *Cluster Center Selection* for *Kmeans* in *DerivedSpace* are most suitable.

4.1 Structure Exploitation

Does a given data set have a structure, or subsets of materials that follow different laws than others?

To answer this question, clusters resulting from applying different clustering methods to the *Work Set* are extracted, and *3d-Multitask-SISSO* (*mSISSO*) is trained on them, with each cluster corresponding to one task. The *Training Set* thus corresponds to the *Work Set*, except for clustering with *HDBSCAN-*λ (see 3.1.1), whose identified outliers are excluded from the *Training Set*. As the derived features are

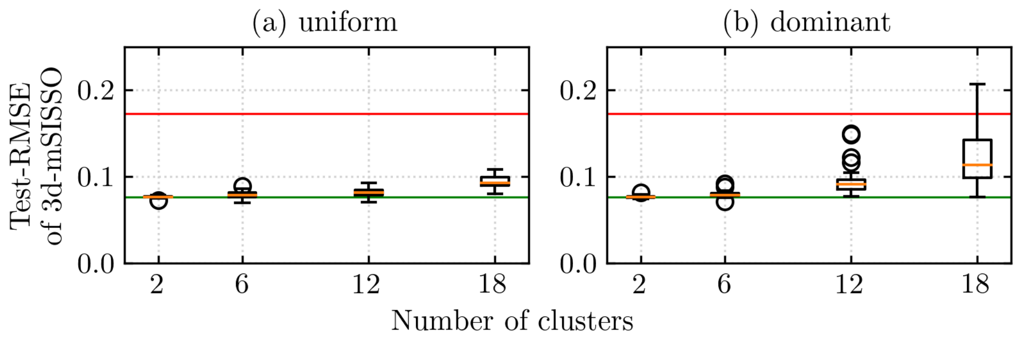


Figure 4.1: Evaluation of whether median test-RMSE of *3d-mSISSO* for random clustering is a valid benchmark for *Structure Exploitation* method assessment. In the left panel (a), training and test materials are uniformly distributed among clusters, while in the right panel (b), a majority of $66,\overline{6}\%$ materials is assigned to one dominant cluster, and the rest is distributed uniformly among the other clusters. In both panels and irrespective of the number of clusters, median test-RMSE is smaller than target property standard deviation (red line), but close to or larger than test-RMSE of *3d-sSISSO* (green line). As median test-RMSEs of the evaluated methods are expected to lay below the latter, it is chosen as benchmark throughout this section.

the same for each cluster, the extracted model is still global, but peculiarities of each cluster are accounted for with their differing coefficients.

As the number of fitting coefficients increases with the number of clusters, it could be assumed that training-RMSE decreases with it irrespective of the quality of chosen cluster structure. The same should not hold for test-RMSE though, as a too fine or otherwise inappropriate clustering probably leads to overfitting. Whether the latter holds is relevant for the choice of a benchmark for method evaluation in this section and is verified for random clustering, whose median test-RMSE of *3d-mSISSO* should thus be close to or surpass test-RMSE of global *3d-sSISSO*. We test this hypothesis for varying *Number of Clusters* c, once for uniformly sized clusters (*uniform*), and once for one dominant cluster containing $66,\overline{6}\%$ of training and test points, and $c-1$ small clusters among which the remaining materials are uniformly distributed (*dominant*). Resulting boxplots of 30 independent runs for each hyperparameter set are shown in fig. 4.1. They are compared to test-RMSE of global *3d-sSISSO* $e^{\text{lat}}_{\text{sSISSO}}$ and target property standard deviation σ^{lat}, which throughout this section are indicated by a green and red line, respectively. Values of the two benchmarks in *Simple Cubic Perovskites* with *Standard Work Set* are $e^{\text{lat}}_{\text{sSISSO}} = 0.077$ and $\sigma^{\text{lat}} = 0.173$. Irrespective of whether the random cluster selection is *uniform* (a) or *dominant* (b) and irrespective of c, median test-RMSEs are close to or surpass $e^{\text{lat}}_{\text{sSISSO}}$, confirming our hypothesis.

As contrary to this an appropriate clustering should decrease median test-RMSE of *3d-mSISSO* as compared to test-RMSE of global *3d-sSISSO* (green line), the latter is thus used as benchmark throughout this section.

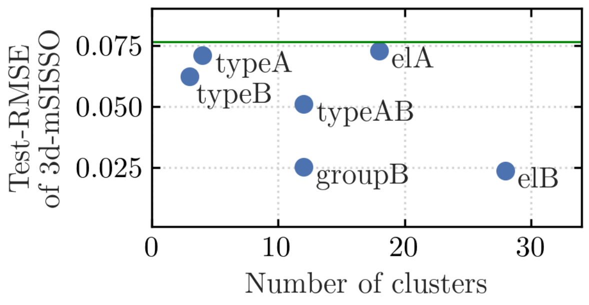


Figure 4.2: Test-RMSE of *3d-mSISSO* for *Simple Cubic Perovskites* with *Standard Test Set* for different clustering methods. For *ElA/B*, materials of same element *A* or *B*, respectively, are assigned to one cluster; for *GroupB*, those whose element *B* belongs to the same column in the periodic table (see fig. 2.1) are ascribed to the same cluster. *TypeA* clustering yields four clusters, where element *A* belongs to the *Alkali Metals*, *Alkaline Earth Metals*, *Transition Metals* or *Lanthanides*, respectively, while for *TypeB* clustering, three clusters are returned containing materials whose element *B* belongs to *Transition Metals*, *Other Metals*, or *Metalloids* as indicated in fig. 2.1. Consequently their combination, *TypeAB* clustering, results in twelve clusters. The green line indicates the benchmark, *3d-sSISSO* test-RMSE.

In the following, results for the different clustering methods as introduced at the beginning of this chapter are presented, followed by a comparative assessment of the clusters and *mSISSO* formula yielded by the best clustering methods.

4.1.1 Element Metadata based Clustering

As can be seen in fig. 4.2, *3d-mSISSO* based on *ElB* clustering yields the best test-RMSE $e^{\mathrm{lat}}_{\mathrm{ElB}} = 0.0239$ with number of clusters $c = 28$, closely followed by *GroupB* with $e^{\mathrm{lat}}_{\mathrm{GroupB}} = 0.0253$ and $c = 12$. Both are a significant improvement as compared to *Work Set* target property standard deviation $\sigma^{\mathrm{lat}} = 0.173$ and *Work Set* trained *3d-sSISSO* test-RMSE $e^{\mathrm{lat}}_{\mathrm{sSISSO}} = 0.077$ (green line). *GroupB* should be preferred over *ElB* as it consists of fewer clusters, and hence achieves the same performance with fewer fitting coefficients.

4.1.2 Clustering in Primary Space

In fig. 4.3 the results for a *3d-mSISSO* application based on *Kmeans* and *HDBSCAN* clustering in *PrimarySpace* for both *noTarget* and *withTarget* version are shown.

As *HDBSCAN* is deterministic, the boxplots indictate the *volatility* of results for seven randomly selected *Test Sets* of same size as the *Standard Test Set*. In order to compare it to the median error of stochastic *Kmeans*, only the *Standard Test Set* results for *HDBSCAN* are used (blue lines).

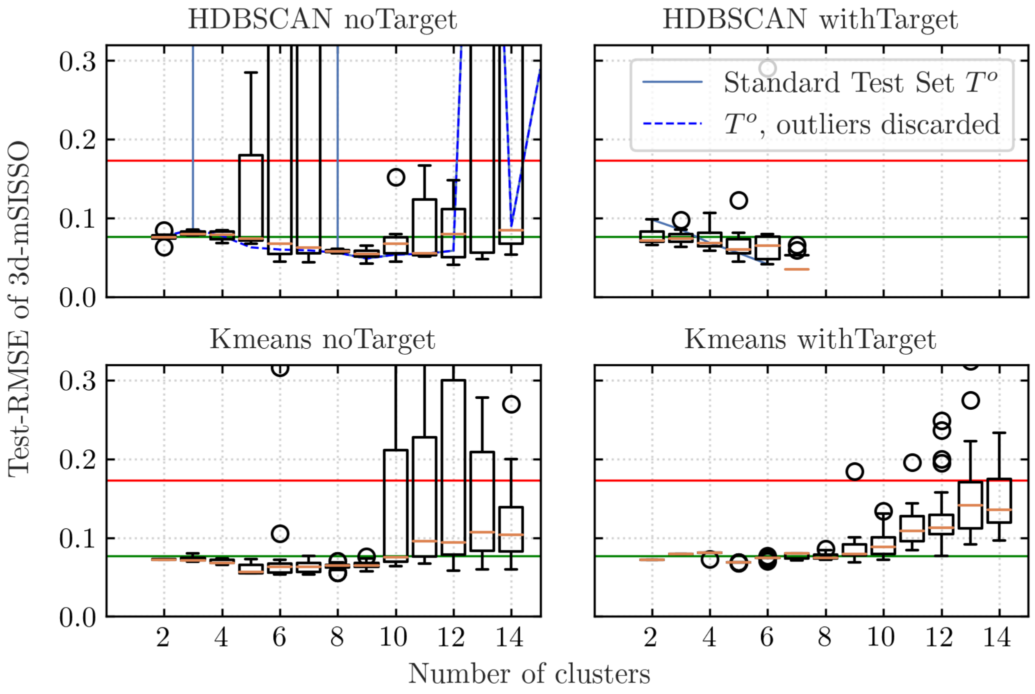


Figure 4.3: Results of different clustering methods in *PrimarySpace*. Target property standard deviation is indicated by a red line, *3d-sSISSO* test-RMSE by a green line. The test-RMSEs of *3d-mSISSO* of *HDBSCAN noTarget* vary highly due to overfitting of a single task for $c \in [4, 7]$. For *Standard Test Set* (blue solid line), additionally discarding tasks as outliers which estimate test point targets outside the confidence interval as indicated in fig. 4.4 results in a smooth curve (blue dashed line) close to the median error. For *Standard Test Set*, both *HDBSCAN noTarget* and *withTarget* seemingly outperform their *Kmeans* counterparts, but at the expense of only describing a subset of the data set as illustrated in fig. 4.5. Comparing the *noTarget* / *withTarget* boxplots of both methods, respectively, shows that including the target for clustering worsens the *mSISSO* estimation.

Although median *HDBSCAN noTarget* errors form a smooth curve with a minimum for $c = 9$, its results are very *volatile*, especially for our *Standard Test Set*. For $c \in [4, 7]$, task 2, which contains one single test point, $SmMnO_3$, exhibits an test error of $e^{\text{lat}}_{\text{task2}} > 1000$, as compared to $e^{\text{lat}}_{task} \in [0.3, 0.9]$ for the other tasks, thus worsening performance considerably. We cannot discard its constituting materials as outliers based on its high task-test-RMSE, as by definition the target property of the *Test Set* is not known. Instead, we can use the target property distribution of *Work Set* materials and its skewed Gaussian fit, which are shown in fig. 4.4, and argue that due to random selection of the *Test Set*, its lattice constant distribution must resemble it. Taking a very conservative confidence interval of 99.999% (whose borders are marked by red dashed lines) allows us to discard test point estimations

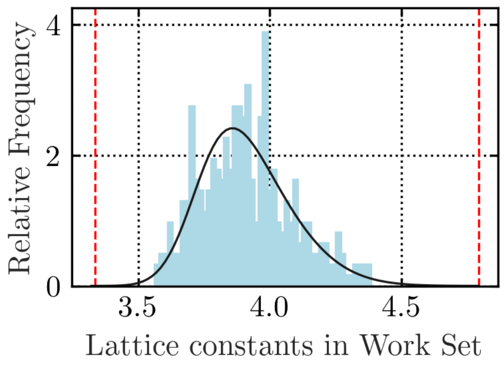


Figure 4.4: Relative frequency histogram of target property (lattice constant) distribution of *Standard Work Set* of *Simple Cubic Perovskites* along with skewed Gaussian fit and its 99.999% confidence interval (red dashed lines). The latter was chosen conservatively in order to avoid discarding extreme, but correct points, tasks or models estimations.

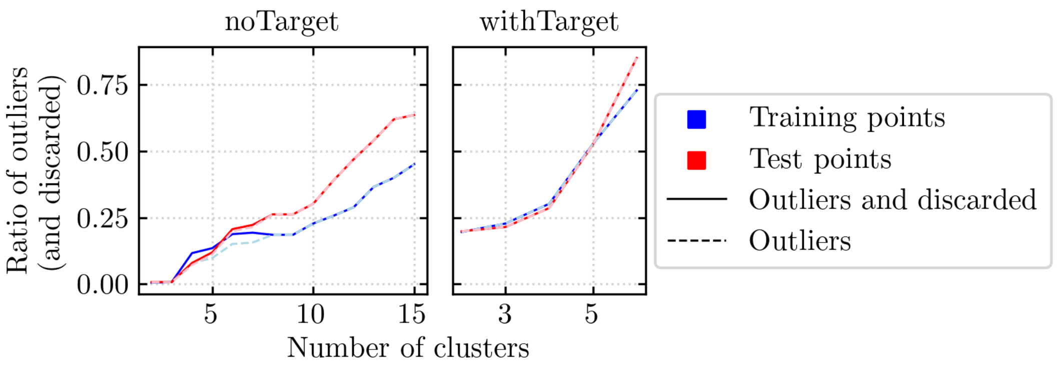


Figure 4.5: Ratio of training and test outliers for *HDBSCAN noTarget* (left) and *withTarget* (right) in *PrimarySpace*. Solid lines represent outliers as discarded by *HDBSCAN-λ* in *Primary Space*, dashed lines account for additionally discarding task 2 due to test point estimations outside of the confidence interval as illustrated in fig. 4.4.

which lay outside of it and their respective tasks / models. Doing so results in discarding $SmMnO_3$ and thus task 2 as outliers and a smooth test-RMSE curve of *3d-mSISSO* for $c \in [4, 7]$, which resembles the median-test-RMSE curve (dashed blue line), and a slight increase of the ratio of *Work Set* and *Test Set* points that are discarded as outliers as shown in fig. 4.5.

Using the approach described above, it appears that all clustering methods result in a smooth curve of *3d-mSISSO* test-RMSE for our *Standard Test Set* that runs close to, but mostly below the *sSISSO* test-RMSE for $c \in [2, 9]$ and then surpasses it when the clustering becomes too fine and overfitting sets in.

Regardless of whether we discard out-of-confidence tasks as outliers, for the *Standard Test Set*, *HDBSCAN noTarget* yields best results at $c = 9$, with $e^{\text{lat}} = 0.054$

with a percentage of discarded outliers as yielded by *HDBSCAN* (see section 3.1.1) of $o_{\mathrm{rel}}^{\mathrm{lat}} = 20\%$ and $o_{\mathrm{rel}}^{\mathrm{lat}} = 26\%$ in the *Work* and *Test Set*, respectively. These data points could be declared to be non-predictable. A better solution though is to fit the whole *Work Set* in the descriptor space spanned by *3d-mSISSO*, thus obtaining a global model that is compatible with the task models, and use resulting fitting coefficients for all outliers. In this case, when the result is assessed, it should be kept in mind that the outliers have not influenced the choice of *mSISSO* descriptor space. The latter might thus not be an optimal model for them, and a possibly increased test-RMSE of outliers as compared to task-test-RMSEs might be attributable to this procedure, and not to an inherent bad predictability of these data points.

The results of *Kmeans noTarget* are comparable to those of *HDBSCAN noTarget*, with median error $m(e^{\mathrm{lat}}) = 0.056$ at $c = 5$ and no outliers. For *Kmeans withTarget*, the best median test-RMSE $m(e^{\mathrm{lat}}) = 0.068$ is yielded for $c = 5$, for *HDBSCAN withTarget* at $c = 6$ with $e^{\mathrm{lat}} = 0.041$ and a percentage of discarded outliers of $o_{\mathrm{rel}}^{\mathrm{lat}} = 73\%$ and $o_{\mathrm{rel}}^{\mathrm{lat}} = 85\%$. This high percentage of outliers suggests that in the *withTarget* version of *PrimarySpace*, materials are scattered and do not cluster at all. As *Kmeans* is outperformed by *TypeB* clustering and the *HDBSCAN* discards a majority of points, the *withTarget* approach in *PrimarySpace* is deemed unsuitable for our purposes and are not investigated further.

4.1.3 Clustering in Derived Space

In fig. 4.6, the test-RMSEs of *mSISSO* based on clustering in *DerivedSpace* are shown.

Compared to their *noTarget PrimarySpace* counterparts, both clustering algorithms yield slightly improved results regarding the best median and IQR of *mSISSO*-test-RMSE. The best (median) error for our *Standard Test Set* for *Kmeans* is $e^{\mathrm{lat}} = 0.0498$ at $c = 8$, for *HDBSCAN* it is $e^{\mathrm{lat}} = 0.0560$ at $c = 7$ and the ratio of points discarded as outliers is significantly reduced to $o_{\mathrm{rel}}^{\mathrm{lat}} = 7\%$ for both *Work* and *Test Set* materials. *DeepAA* clustering does not yield any promising results along the entire c range.

Regardless of the number of clusters, the results based on *DeepAA* clustering are worse than those of *sSISSO*. This might be explained by its original purpose, namely to derive a *Latent Space* or *Derived Space* which is not only optimized for estimating the target property, but also for reconstructing primary features. For its original purpose of artificially generating pictures of faces/ structures of molecules (our primary features) with a specific emotional expression/ heat capacity (our target property) [3] for any point in latent space, the ability to reconstruct both primary features and target property is central. For our application to materials science data though, simultaneously optimizing target property and primary feature reconstruction is not necessary nor useful, as is expounded in the following:

For pictures, the value on the continuous grey-scale of each bit in the bitmap is largely independent of the values of all other bits. Contrary to that, for the investigated *Simple Cubic Perovskites* data set, the corresponding values for all material primary features are determined by the elements that constitute them. Possible values are thus discrete, and the ones of each feature of the same element

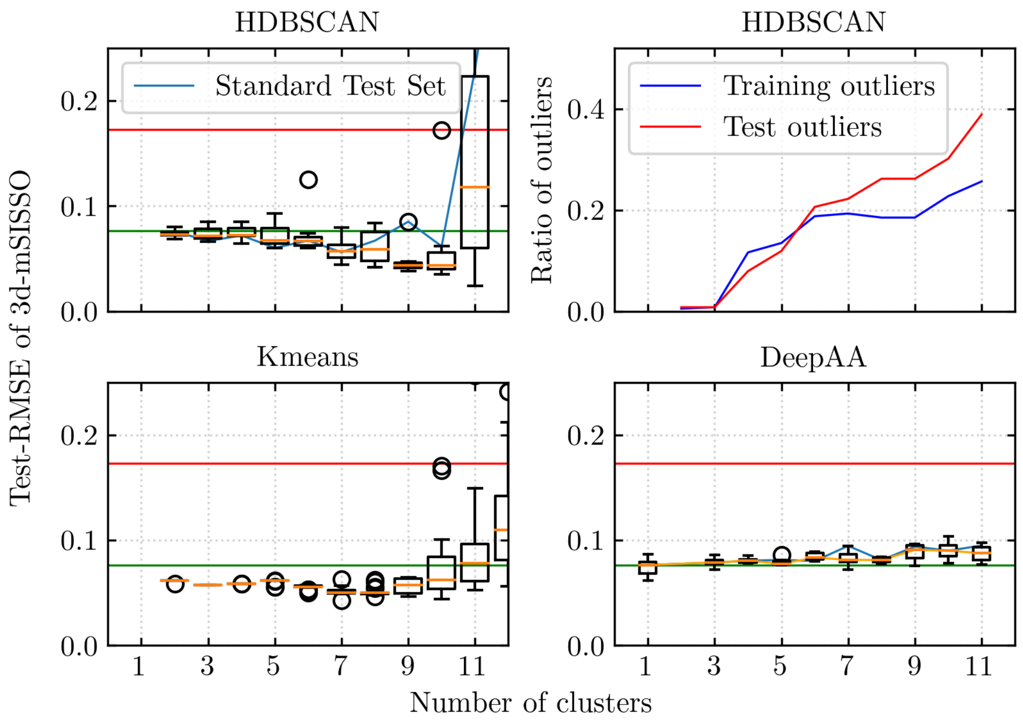


Figure 4.6: Results of different clustering methods in *DerivedSpace*. Target property standard deviation is indicated by a red line, *3d-sSISSO* test-RMSE by a green line. Contrary to its *PrimarySpace noTarget* version, *Standard Test Set* test-RMSEs for *mSISSO* based on *HDBSCAN* clustering in *DerivedSpace* (top left) decrease smoothly with increasing c until overfitting starts at $c = 11$. Also, the ratio of outliers discarded by *HDBSCAN* (top right) is much smaller for the relevant range of c and justifiable. The best test-RMSE yielded for *Kmeans* clustering in *DerivedSpace* (bottom left) slightly outperformes the one of clustering in *PrimarySpace* and *noTarget*. *DeepAA* (bottom right) does not show any promising results on the entire c range.

(A or B) are completely dependent. For more complex data sets with variable stoichiometry and/or more than two elements, primary features are often represented as the weighted mean and standard deviation of the material's constituting elements. Allowing for off-stoichiometric compositions, each individual feature value is thus continuous- still, they are determined by the constituting elements and composition ratios and are thus completely dependent. Due to this dependency, the *(Reconstructed) Real Space* spanned by these features is also only discretely populated by points or hyperplanes representing possible actual materials.

Estimated feature values scatter in *Reconstructed Real Space* as the strict feature dependence can only indirectly be addressed by *DeepAA* via the feature reconstruction error. Selecting any unknown point in *Latent Space*, it is thus unclear whether the reconstructed primary features match. The latter determines if constituting elements and composition ratios can be derived from them, hence, whether the re-

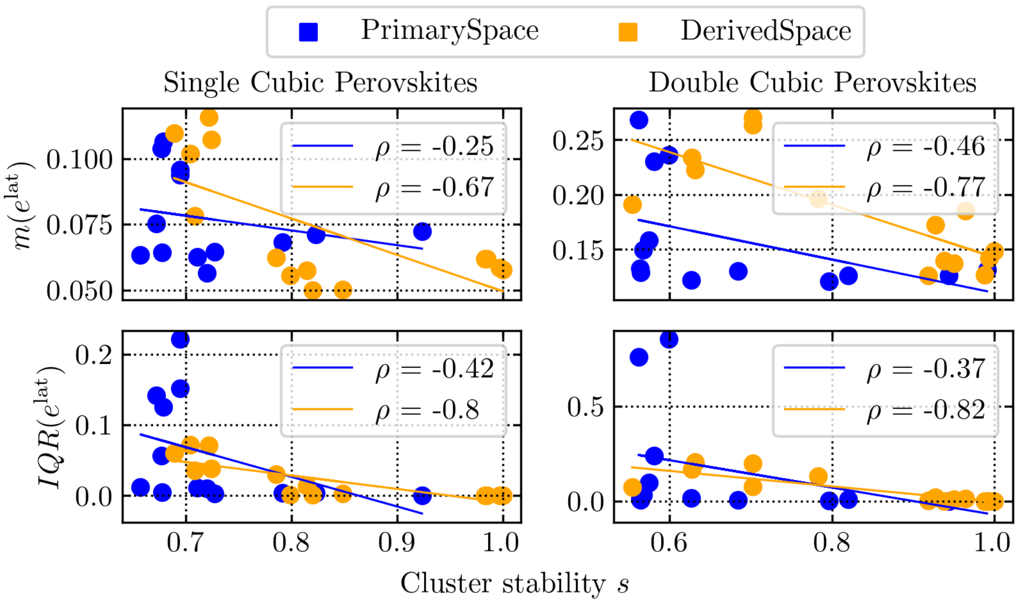


Figure 4.7: Fit of Pearson correlation coefficient ρ of median (top) and IQR (bottom) of test-RMSE e^{lat} to *Cluster Stability s* for *3d-mSISSO* based on *Kmeans noTarget* in *PrimarySpace* and *Kmeans* in *Derived Space* for *Single* and *Double Cubic Perovskites.*

constructed data point represents a possible actual material. At the same time, reconstruction is not necessary as we can navigate *Latent Space* and corresponding target value with the trained model by feeding the *Real Space* branch all possible constituting elements and composition ratios combinations- a method which does not apply to pictures of faces that have not been taken yet nor for the huge number of possible complex molecule structures.

The reconstruction branch of *DeepAA* is thus not useful for our purposes and could be omitted, as simultaneously optimizing target property and primary feature reconstruction yields worse results than optimizing target property reconstruction only. An adapted neural network without the feature reconstruction branch and feature reconstruction error penalty term, all other things being equal, would still lead to a *Latent Space* and thus also clustering which comprises information about both primary features and target value, comparable to *sSISSO*'s *Derived Space.*

For future work, an assessment of whether the architecture and objective function of the algorithm candidate are suitable for the task at hand should be conducted before implementation.

4.1.4 Assessment of Best Structure Exploitation Methods

As can be seen in fig. 4.3, median test-RMSE resulting from *Kmeans noTarget* in *Primary Space* and *Kmeans* in *DerivedSpace* is lowest where its IQR is lowest. Subsequently it is assessed whether both in turn are determined by the *Cluster Stability* of the stochastic *Kmeans* for given *Number of Clusters c*. We define *Cluster Stability* s_c as the average *Cluster Similarity* $s_c^{i,j}$ over all iteration pairs. The latter we define by the Jaccard index[1] [13] of the two sets of same-cluster-membership *Work Set* material pairs P^i, P^j for two iterations $i, j \in \{1, 2, ..., 30\}$:

$$s_c^{i,j} = \frac{|P^i \cup P^j|}{|P^i \cap P^j|} \tag{4.1}$$

$$s_c = \overline{s_c^{i,j}} \tag{4.2}$$

Both $s_c^{i,j}$ and s_c range from zero to one, with a higher value indicating more similar clusters and a more stable result of the stochastic *Kmeans*, respectively. The results are displayed in fig. 4.7. Each point represents the *Cluster Stability* s_c and median (left) or IQR (right) *3d-mSISSO* test-RMSE for a given number of clusters c for *Kmeans noTarget* in *PrimarySpace* and *Kmeans* in *DerivedSpace*.

As we are not interested in the amount of average influence s has on the median or IQR, respectively, which would correspond to the slope of a linear fit, but in the question whether s and the latter two are (linearly) related, and if so, how noisy that relationship is, the Pearson correlation coefficient is applied.

For *Kmeans noTarget* in *PrimarySpace*, a weak negative Pearson correlation coefficient results for both median ($\rho = -0.25$) and IQR ($\rho = -0.42$), while for *Kmeans* in *DerivedSpace*, they are quite strong ($\rho = -0.67$ and $\rho = -0.8$, respectively). Hence, only the stability of *Kmeans* clustering in *DerivedSpace* might be usable as an indicator for the potential for error improvement and *error volatility*. Cluster stability in term is directly determined by the distinctiveness of the cluster structure, and hence of the heterogeneity of the data set. In order to confirm whether *Cluster Stability* is a suitable indicator, this analysis is repeated with *Double Cubic Perovskites*, which contains *Single Cubic Perovskites* but exhibits increased heterogeneity.

Again, clustering in *PrimarySpace* results in rather weak correlation coefficients of $\rho_{s,m} = -0.46$ and $\rho_{s,\text{iqr}} = -0.37$ and clustering in *DerivedSpace* in strong correlations of $\rho_{s,m} = -0.87$ and $\rho_{s,\text{iqr}} = -0.93$. This supports the hypothesis that *Kmeans* clustering in *DerivedSpace* is a more suitable indicator for error improvement than in *PrimarySpace*.

In fig. 4.8, the clusters yielding best test-RMSE are illustrated. The vertical axis represents the *set A*, the horizontal one the *set B* element of the respective material. The first of the two elements indicates the top row/ left column, the second one the bottom row/right column, respectively. Cell color denotes material cluster membership. White indicates *Test Set* materials, grey outliers. Clustering *ElB* is not displayed.

It can be seen that the most successful clustering, *GroupB*, yields clusters which are not found by *Kmeans* and *HDBSCAN*. An exception is the assignment of Zn and

[1] Akin to Tanimoto similarity [12].

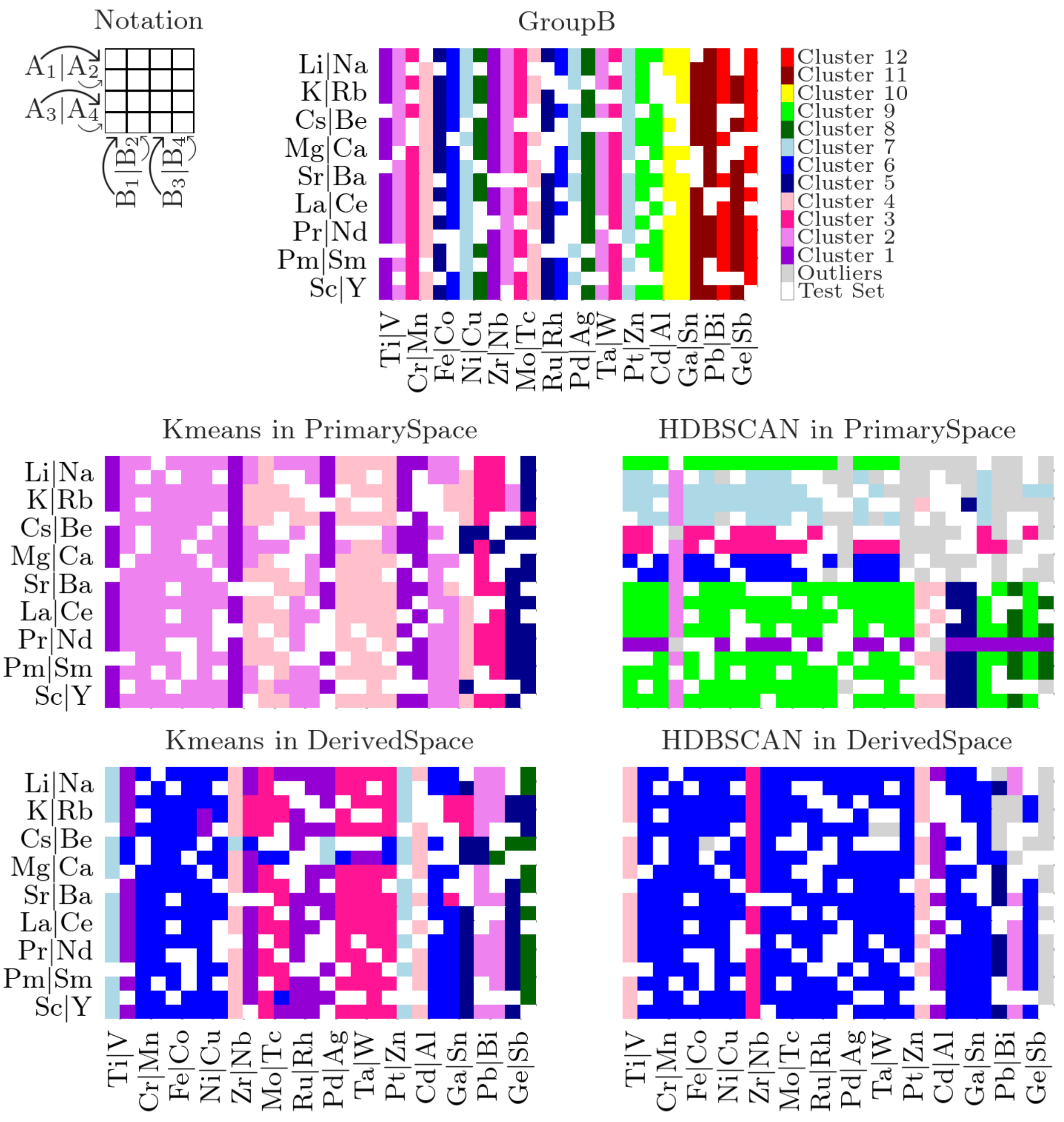


Figure 4.8: Clustering resulting in best test-RMSE for *Standard Test Set*. Element denotation X/Y indicates that X is the element corresponding to the upper row/ left column, thus Y to the lower row/ right column of its two neighboring slices. Cluster membership is indicated by color, outliers by grey and *Standard Test Set* by white cells. Resulting *Number of Clusters* are $c = 5$ and $c = 7$ for *Kmeans* and *HDBSCAN noTarget* in *PrimarySpace*, respectively, and $c = 8$ and $c = 6$ for *Kmeans* and *HDBSCAN* in *DerivedSpace*.

Cd to the same cluster, which is also the result of both methods in *PrimarySpace* and assigning Al and Ga to the same cluster, which is found by all methods. Interestingly,

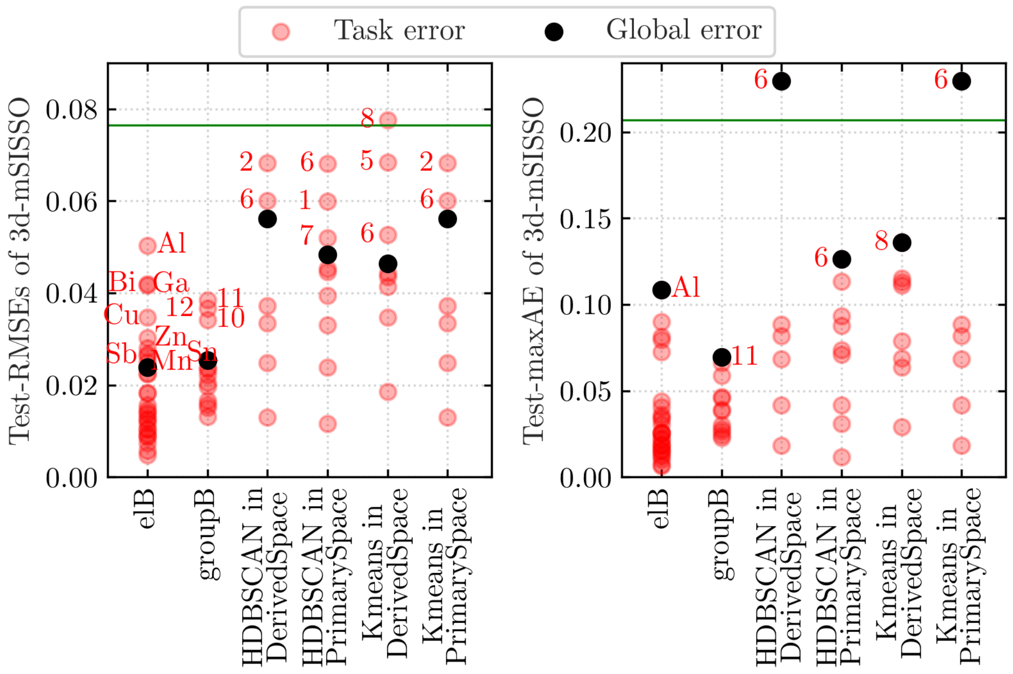


Figure 4.9: Task and global errors resulting of best-test-RMSE yielding clustering methods. The green line indicates the benchmark, *3d-sSISSO* test-RMSE. The letters correspond to the task assignments shown in fig. 4.8, except for *ElB* clustering tasks, whose set *B* element is annotated. Except for test-RMSE of task *H* for *Kmeans* in *DerivedSpace* and MaxAE of task *F* for *HDBSCAN* in *DerivedSpace*, which are slightly higher than the respective errors of *sSISSO*, all task-errors are consistently smaller.

despite *sSISSO*s nonlinear space transformation and different optimal number of clusters *c*, the structures found by *Kmeans noTarget* in *PrimarySpace* and *Kmeans* in *DerivedSpace* are quite similar: They yield many overlapping material groups of same-cluster-membership, which for each method are combined differently to clusters: One group contains *set B* elements Cr to Cu and Al and Ga, another one elements Ag to Pt plus some Ru and Rh materials, a third group Pb and Bi elements and for both methods, all Ti and Zr materials belong to the same group, respectively.

The latter two groups are also found by *HDBSCAN* in *DerivedSpace* which additionally shares the Zn, Cd and Ge group, respectively, with *Kmeans* in *DerivedSpace*, to which it is thus more similar than to *HDBSCAN* in *PrimarySpace*. The latter is the only one that also extracts four clusters based on set *A* elements, namely Na to Cs, Be and Mg, Ca and Sr and the Nd cluster.

For the assessment of the resulting data set structure and its task performance, both RMSE and maximum absolute error (MaxAE) of target property estimation for each task in the *Test Set* are valuable: The former measures the prediction performance of the model for the whole task, which, regarding all tasks jointly, is

Table 4.1: Derived features $d_1, ..., d_3$ of *3d-mSISSO* model lat $= c_0^t + c_1^t d_1 + c_2^t d_2 + c_3^t d_3$ with tasks $t = 1, ..., T$ for best-test-RMSE yielding clustering methods. *Prim.* ✕ stands for *noTarget PrimarySpace*, *Der.* for *DerivedSpace*.

Clustering Space		d_1	d_2	d_3
global model		${r_{sA}}^{Z_B/EN_B}$	$r_{sB} \exp^{EA_B}$	$IP_A \cdot {}^{EA_B}/{r^3_{pB}}$
ElB		$\frac{\sqrt{Z_A}}{IP_A + HOMO_A}$	$\log(Z_A)(IP_A - LU_A)$	$\frac{EA_A - EN_A}{\lvert EA_A - HO_A \rvert}$
GroupB		$\frac{LU_B}{Z_B \sqrt[3]{r_{sA}}}$	$\frac{\sqrt[3]{Z_A} EA_B}{r_{sA}}$	$\frac{\sqrt{Z_A}}{IP_A + HO_A}$
Prim. ✕	*Kmeans*	$\frac{Z_B}{EN_B \cdot r_{pB}}$	$\exp\left(r^3_{sB}\right)$	$(EA_B - HO_B)\exp(r_{sA})$
	HDBSC.	$\frac{EN_A - EN_B}{Z_B \cdot r_{sA}}$	$\frac{HO_A \cdot LU_B}{\sqrt[3]{EA_B}}$	$Z_B^2 \log(r_{sB})$
Der.	*Kmeans*	$Z_B \frac{r_{sA} + r_{sB}}{IP_B}$	$\frac{Z_A}{r_{pB}(IP_A + HO_A)}$	$\lvert EA_B - EN_B - HO_B \rvert$
	HDBSC.	$\frac{Z_A}{IP_A} + \frac{Z_B}{EN_B}$	$\frac{\sqrt[3]{EA_B}}{Z_B r_{sA}}$	$\frac{IP_A}{HO_B}(IP_B + HO_A)$

an indicator of the overall meaningfulness of the found structure, while the latter focuses on the single task point of worst prediction, thus measuring how "clear-cut" the found clusters are. Hence, for all best clustering methods discussed above, both error types of each task (corresponding to one cluster, respectively) are displayed in fig. 4.9. Almost all exhibit smaller test-RMSE and MaxAE then the simple *sSISSO* application, except for task H of *Kmeans* in *DerivedSpace* and task F of *HDBSCAN* in *DerivedSpace*, whose RMSE and MaxAE, respectively, are slightly higher. Although *GroupB* clustering results in a slightly higher global test-RMSE than *ElB* clustering, its variance of task-test-RMSEs is significantly lower. The same holds for its global test-MaxAE and the variance of test-MaxAE amongst its tasks. Furthermore, as it relies on far less clusters, we consider *GroupB* superior to *ElB* clustering.

The partial similarity of the cluster structures for different clustering methods as displayed in fig. 4.8 is not reflected in the derived features yielded by *mSISSO* (see table 4.1). This is notable especially for the two *mSISSO* models based on clustering in *DerivedSpace*, whose derived features do not resemble the ones of *sSISSO*. Although the *mSISSO* estimations are thus reliable globally as well as for each task and each point, human-readable and reusable, derived feature variety in models of similar quality suggests that its solutions are not unique.

As was to be expected, for *ElB* clustering, the variance in target property caused by the features of element B is not reflected in the human-readable, analytical derived features d_i for $i = 1, ..., 3$, which only depend on features of element A. The influence of element B features is thus entirely captured in the varying task coefficients c_i^t for $i = 0, ..., 3$, whose values are not mapped to primary features by analytical formula. In the following, a short formalization of this feature of *mSISSO* is conducted:

Generally, the predictive power of a statistical model can be assessed via the *Coefficient of Determination* [14] $R^2 = 1 - \frac{SS_{\text{res}}}{SS_{\text{tot}}}$. Here, $SS_{\text{res}} = \sum_i (y_i - f_i)^2 = \sum e_i^2$

denotes the residual sum of squares, f_i the prediction of model f for point i and e_i its residual or error, and $SS_{\text{tot}} = \sum_i (y_i - \overline{y})^2$ the total sum of squares, which is proportional to target property variance. It thus measures the performance of model f as compared to the one of simply taking the target property mean $\overline{y}$. For *mSISSO*, an analogous coefficient can be constructed which captures the ratio of residual sum of squares reduction we obtain by allowing for differing coefficients in the global linear model $\overline{f}$ in the given descriptor space for each cluster c, thus obtaining f_c: The value of the *Coefficient of Determination* via *Coefficient Variation* $C^2 = 1 - \frac{\sum(y_i - f_{c,i})^2}{\sum(y_i - \overline{f}_i)^2}$ indicates both the meaningfulness of the underlying cluster structure for target property prediction, and the amount of variance which is captured by the model through varying coefficients, but not derived from primary features in form of analytical physical formula.

For *ElB* clustering, all possible influences of element B features on the task coefficients c_i^t are thus "hidden" in this representation and can themselves be estimated with *sSISSO* in order to yield a global model.

In an attempt to obtain such a global model of similar quality as measured in test-RMSE, *sSISSO* with maximum dimension $d_{\max} = 3$ and $rung = 2$ is run four times independently with the respective coefficient c_i^t as target property for $i \in \{0, 1, 2, 3\}$ with $t \in ElB$ and the respective element B features of *Simple Cubic Perovskites* as features. All elements are used in the *Training Set* as for each, we have all information about both target properties and features. For all coefficients, *3d-sSISSO* models yield lowest RMSE. Despite the high model complexity of a three dimensional model, their coefficient predictions $\tilde{c_i}^t$ are thus chosen as a replacement for the ones yielded by *mSISSO*. This eliminates the unexplained task differences and yields a global model for lattice constant prediction, whose test-RMSE though is significantly increased to $e^{\text{lat}} = 0.0586$ as compared to $e^{\text{lat}}_{\text{ElB}} = 0.0239$. Furthermore, as this process leads to an increased effective $rung = 3$ and $d = 12$ of the lattice constant prediction model, it is not investigated further.

4.2 Equivalent Reduced Data Set

Can the data set be reduced substantially, such that training the model still yields results of similar quality, and are there stable or unique data points whose inclusion during training is strictly necessary in order to obtain such a model?

To address this question, the clustering-based *Training Set* selections are compared to a completely random selection (see fig. 4.10). Light blue solid and dashed lines represent the median and IQR of *3d-sSISSO* test-RMSEs for 30 random *Training Set* selections and *Standard Test Set*, respectively. Due to their high *volatility*, their smoothed counterparts (blue lines) are used as benchmarks in their place, which are calculated as the moving average of their ± 10 neighbors.

It is to be expected that for any clustering method M, selecting a *Training Set* from the 378 materials containing *Standard Work Set* yields a model whose median test-RMSE $m^{M,n}(e)$ is at best (approximately) as large as the one for *sSISSO* trained on the whole *Work Set* $e^{\text{Work}} = 0.077$, or slightly below. We thus measure the success of method M by *Normalized Median Error Increase*

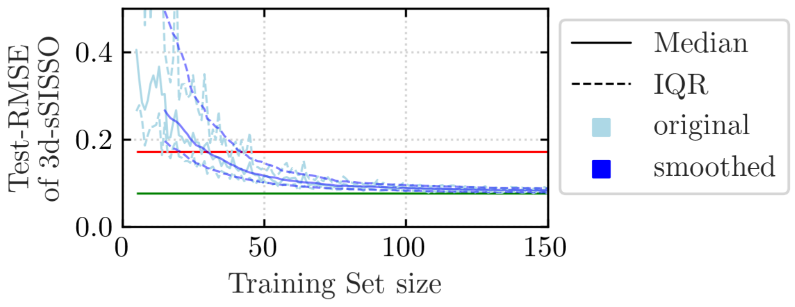


Figure 4.10: Random *Training Set* selection benchmark: For each *Training Set* size, a random subset of the *Standard Work Set* is drawn 30 times, whose median and IQR are displayed in light blue. Due to their high *volatility* especially for smaller *Training Set* sizes, the smoothed median and IQR (blue solid and dashed lines, respectively) serve as benchmarks instead. Target property standard deviation is indicated by a red line, *3d-sSISSO* test-RMSE by a green line.

$$nmei(e^{M,n}) = \frac{m(e^{M,n}) - e^{\text{Work}}}{m^s(e^{r,n}) - e^{\text{Work}}} \tag{4.3}$$

with $m^s(e^{r,n})$ being the smoothed median test-RMSE for random *Training Set* selection of same size n. As $m^s(e^{r,n}) - e^{\text{Work}} > 0$ always holds, $nmei > 1$ indicates that the error of *Training Set* selection method M is worse than for a randomly selected *Training Set* of size n. For $0 \leq nmei \leq 1$, method M is superior to random selection, for $nmei < 0$ it is even better than training *sSISSO* on the whole *Work Set*, which is expected to not- or only rarely- be achieved.

We also define *Normalized IQR*

$$nIQR(e^{M,n}) = IQR(e^{M,n})/IQR(e^{r,n}) \tag{4.4}$$

as the ratio of IQR resulting from method M and the one yielded by random selection for n train points, respectively. Where *Normalized Median Error Increases* indicate a successful method, $nIQR \leq 1$ ensures that an improvement of median error is not achieved at the expense of (substantially) increased *error volatility*.

In the following, two methods for *Training Set* selection are regarded - the *Cluster Center* and *(Guided) Random Selection* approach.

4.2.1 Cluster Center Selection - in Primary and Derived Space

The first approach - *Cluster Center Selection* - chooses the material of lowest Euclidean distance to its cluster center, whose position in turn is calculated as the average of the position of all cluster members. This *Selection Method* thus results in a *Training Set* size that equals the number of clusters c. It cannot be applied in neither of the *TypeA/B/AB*, *GroupA/B* nor *ElA/B* approaches as in the "spaces" spanned by type, group and element, measuring the Euclidean distance is not possible. As for

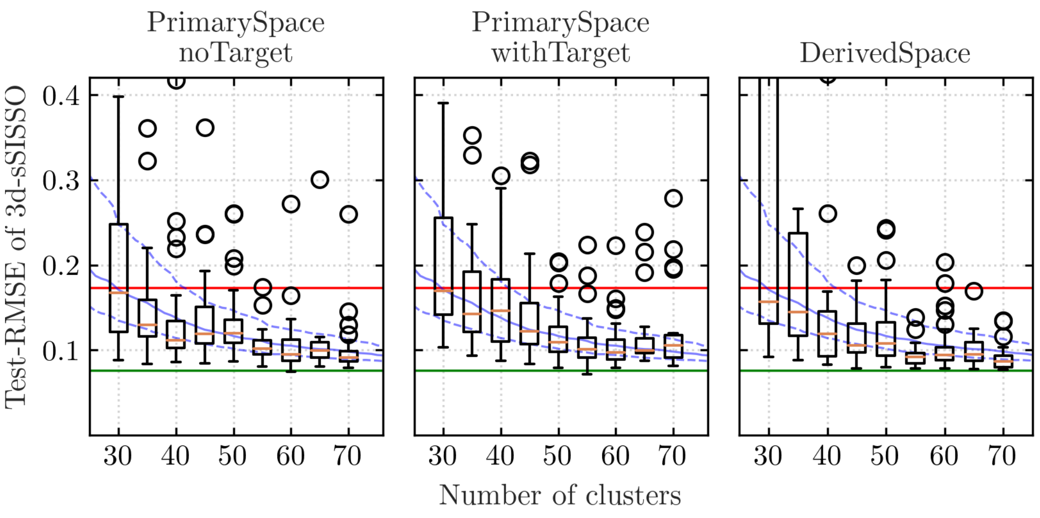


Figure 4.11: Boxplots of *3d-sSISSO* test-RMSE based on 30 *Training Sets* of size c, each consisting of the materials closest to the cluster centers for c clusters, as yielded by non-deterministic *Kmeans* in *PrimarySpace* for *noTarget* (left) and *withTarget* (middle) and *Kmeans* in *DerivedSpace* (right) as compared to smoothed random-selection median (solid blue line) and IQR-borders (dashed blue line). Target property standard deviation is indicated by a red, *3d-sSISSO* test-RMSE by a green line.

HDBSCAN and *DeepAA* the maximum number of clusters c_{max} is limited, for the former by the number of leafs in the dendogram, and for the latter by the number of primary features, which results in a small $c_{max} \leq 25$ for both algorithms, the *Cluster Center* approach is only applied to *Kmeans*. In fig. 4.11, the results of 30 *Kmeans* applications in *noTarget* or *withTarget PrimarySpace* and in *DerivedSpace* are displayed and compared to the smoothed median and IQR for random selection. For *noTarget PrimarySpace*, the medians and IQR borders lay consistently below their random counterparts except for $c = 50$, with an average $\overline{nmei}(e^{\text{noTarget}}) = 0.770$. For *withTarget PrimarySpace*, the same holds only for its medians, with exception of $c \in \{20, 40, 70\}$, with an average $\overline{nmei}(e^{\text{withTarget}}) = 0.976$. As before, the *noTarget* is superior to the *withTarget* approach, which yields only marginal improvements to random subset selection. In *DerivedSpace*, $\overline{nmei}(e^{\text{Der.Sp.}}) = 0.696$, outperforming *PrimarySpace* applications. For few clusters though, $c = 30$, *error volatility* tripled, $nIQR(e^{\text{Der.Sp.},30}) = 3.056$; for all other method M and n_{train} combinations, $nIQR(e^{M,n_{\text{train}}}) \lessapprox 1$.

4.2.2 Guided Random Selection - Element Metadata based Clustering

In order to avoid disregarding one chemical element completely and obtaining a high *error volatility* for small *Training Set* sizes, the random subset selection based on element metadata clustering is partially guided. For element metadata based clustering methods, in each cluster the same number of materials $n_{\text{perCluster}}$ are selected with

Table 4.2: Mean *Normalized Median Error Increase* and mean *nIQR* for different element metadata based clustering methods.

	ElA	ElB	GroupA	GroupB	TypeA	TypeB
$\overline{nmei}$	-1.567	0.601	0.310	0.859	0.655	1.076
$\overline{nIQR}$	1.053	0.823	1.284	0.627	0.761	1.017

the following algorithm: Shuffle the sequence of clusters. For each cluster, iteratively randomly select a material from the ones in the cluster whose elements have been least picked in the already chosen *Training Set* materials. Repeat this $n_{\text{perCluster}}$ times, hence yielding $n = c \cdot n_{\text{perCluster}}$ materials that are evenly distributed among clusters and exhibit the most uniform element A and B distributions possible. The upper border for $n_{\text{perCluster}}$ is given by the minimum number of *Work Set* materials for all clusters. Thus, *TypeAB* clustering is not applicable with this selection algorithm because $n_{\text{perCluster}} = 2$ for the Sc-Y/Ga-Sb cluster as the materials $ScGeO_3$ and $YSbO_3$ are part of the *Test Set*, hence yielding $\max(n_{\text{total}}) = 24$.

This procedure is thus applied 30 times based on *ElA/B*, *GroupA/B* and *TypeA/B* clustering - in table 4.2, mean *Normalized Median Error Increase* and mean *nIQR* are displayed. All methods except *TypeB* exhibit a mean $\overline{nmei}$ below 1, with substantial mean median improvements of approximately 33, 40 and 69% for *TypeA*, *ElB* and *GroupA* clustering, respectively, and even a mean median test-RMSE that lays below *sSISSO* test-RMSE when trained on the whole *Work Set* for *ElA* clustering. Mean *nIQR* are smaller than one except for *ElA* and *TypeB* clustering, for which it is slightly higher, and *GroupA*, for which it is substantially bigger than one.

A more detailed view of $nmei(e^{M,n})$ and $nIQR(e^{M,n})$ is provided with fig. 4.12. On the left hand side, *Normalized Median Error Increases* are displayed, alongside a red line at $nmei = 1$ above which medians resulting after clustering are worse than those resulting from random selection, and a green line at $nmei = 0$, below which median errors are smaller than *Work Set* trained *3d-sSISSO* test-RMSE. On the right hand side, the red line at $nIQR = 1$ indicates where *error volatility* of the method surpasses its random counterpart.

For all methods except *GroupB* and *TypeB*, which surpass the red border multiple times for n_{train} bigger 168 and 39, respectively, medians resulting from guided random selection inside the clusters lay consistently below the one of random selection, and *nmei* decreases with increasing n_{train}. For *GroupA* and *ElA* clustering, for n_{train} bigger 288 and 112, respectively, median error even lays consistently below *Work Set* trained *3d-sSISSO* test-RMSE. Also, all methods exhibit a *nIQR* consistently below 1, except for *ElA*, *GroupA* and *TypeB*, whose *nIQR* is bigger for n_{train} regions corresponding to close to or outside of $nmei = 0$ and $nmei = 1$ borders. Interestingly, *TypeA* and *GroupA* *nmei*'s differ significantly, especially for $n_{\text{train}} \geq 112$, although the latter clustering corresponds to the former, except that every lanthanide element forms its own cluster, with exception of La, which is assigned to the Sc and Y containing cluster (see fig. 2.1). Thus, a *Training Set* selection that contains disproportionately many Ce-, Pr-, Nd-, Pm- and Sm-containing materials seems to yield better results for $n_{\text{train}} \geq 112$. Please note though that for both *ElA* and *GroupA*,

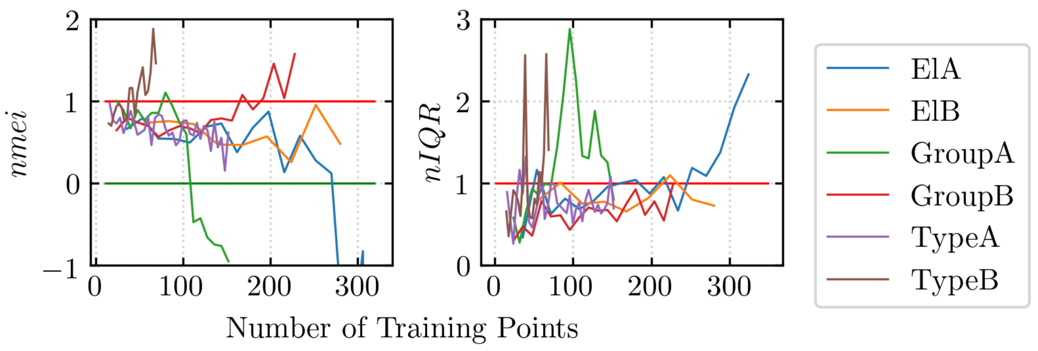


Figure 4.12: *Normalized Median Error Increase* (left) and *nIQR* (right) for guided random selection based on different element metadata based clusterings. Above the red lines at $nmei = 1$ and $nIQR = 1$, respectively, medians and IQRs resulting from the corresponding method are higher than their random selection counterparts, and below the green line at $nmei = 0$, medians yielded by the method are smaller than test-RMSE of *3d-sSISSO* trained on the *Work Set*.

for $nmei < 0$ absolute median errors are still consistently $m(e) > 0.7$, and thus close to $e_{\text{sSISSO}}^{\text{WorkSet}} = 0.077$.

4.2.3 Random Selection - Clustering in Primary Space

For *Kmeans*, *HDBSCAN* and *DeepAA*, conducting a selection of points as expounded above proves to be unfeasible, as for these algorithms clusters of significantly differing sizes appear; trying to extract the same number of points from each cluster would firstly drastically limit the number of total points selected, $n_{\text{total}} = n_{\text{perCluster}} \cdot c$, as $n_{\text{perCluster}}$ is limited by the smallest cluster, and secondly disproportionately assign weight to smaller clusters when fitting the final *sSISSO* model.

The method is thus adapted to select, for a given train ratio r^{train}, the corresponding rounded number of materials in each cluster randomly, but at least one, in order to avoid disregarding smaller clusters completely. Due to varying cluster sizes resulting from non-deterministic *Kmeans*, this algorithm can yield different total numbers of materials $n_{\text{total}} = \sum_{i=1}^{c} \max(1, \lfloor r^{\text{train}} n_i \rfloor)$ for the same r^{train} and c. For a given number of clusters c and train ratio r^{train}, non-deterministic *Kmeans* both *noTarget* and *withTarget* are run 30 times in *PrimarySpace* and random selection is conducted once on each of them. Deterministic *HDBSCAN-λ* in *PrimarySpace* is run once for *noTarget* and *withTarget* and resulting clusters are selected 30 times randomly. Due to the additional dependence on hyperparameter c the assessment of their results is more complex. For that, in fig. 4.13, *Normalized Median Error Increase* $nmei(e^{M,c,r^{\text{train}}})$ are regarded as well as their means $\overline{nmei}(e^{M,c})$, $\overline{nmei}(e^{M,r^{\text{train}}})$ and $\overline{nmei}$ in order to make visible possible trends and compare methods. Cells whose median test error exceeds standard error of the target property are marked by a red cross, as corresponding models are not useful for target property prediction.

Similar to the *Structure Exploitation* results of *mSISSO*, *noTarget* clustering slightly

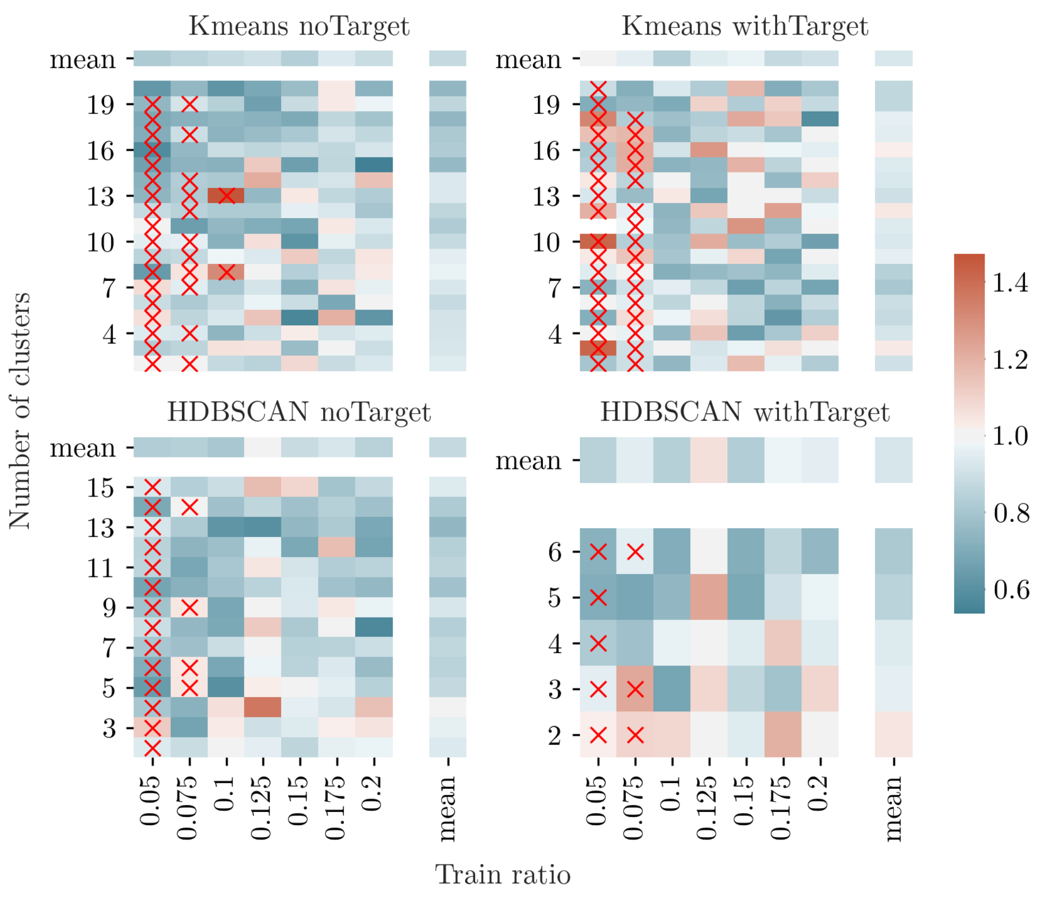


Figure 4.13: *Normalized Median Error Increase* (*nmei*) for *Kmeans* and *HDBSCAN* in *Primary Space* for *noTarget* and *withTarget* for varying number of clusters c and train ratios r^{train} as well as their mean values for fixed c or r^{train}. Red crosses indicate that the cell's corresponding median test error exceeds the standard deviation of the target property.

outperforms *withTarget* clustering, with $\overline{nmei}^{\text{Kmeans}} = 0.862$, $\overline{nmei}^{\text{HDBSCAN}} = 0.890$ for the former and $\overline{nmei}^{\text{Kmeans}} = 0.916$, $\overline{nmei}^{\text{HDBSCAN}} = 0.915$ for the latter. The reason for that differs though: For *mSISSO* based on *withTarget* clustering, assigning cluster memberships to test points relies on by definition unknown target properties, which were thus estimated with *sSISSO*. Resulting faulty cluster assignments of some test points due to limited estimation accuracy might have led to the inferior performance of *withTarget* to *noTarget* based *mSISSO*. For the random selection approach, such a test point cluster membership determination is not necessary. Here, the *withTarget* approach in *PrimarySpace* simply leads to selection which weighs the importance of a target property distribution resemblance of *Training* and *Work Set* equal to the one of their joint primary feature distributions. The better performing *noTarget* version in contrast leads to a selection for which only a primary features distribution resemblance of *Training* and *Work Set* results, which thus seems to be

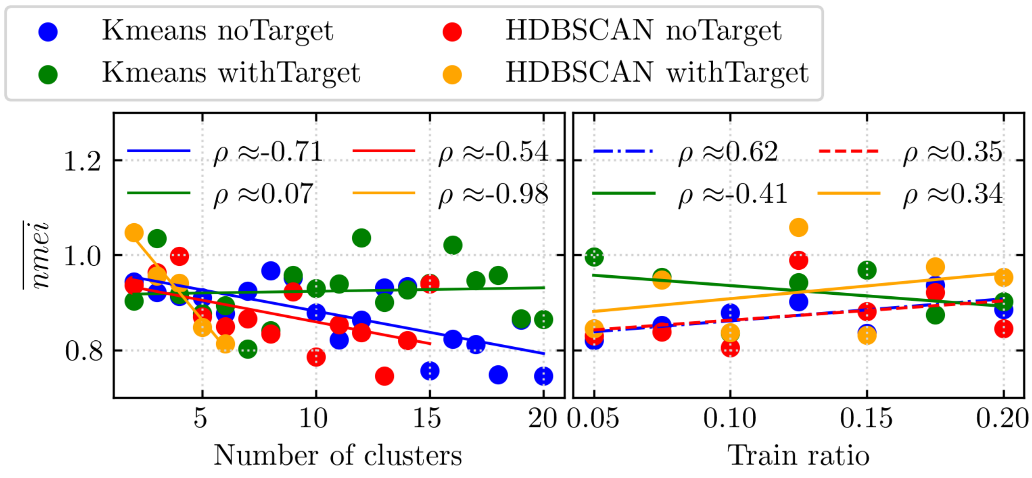


Figure 4.14: Pearson correlation coefficients of *Normalized Median Error Increase* (*nmei*) with number of clusters c and train ratio r^{train} for *Kmeans* and *HDBSCAN* in *PrimarySpace* for *noTarget* and *withTarget*. Except for *Kmeans withTarget* strong negative correlations with c are yielded, $\rho_{c,nmei} \in [-1, -0.5]$, and weak positive correlations with r^{train} with $\rho_{r^{\text{train}},nmei} \in [0, 0.5]$.

more important for finding *Representatives* of a data set.

In fig. 4.14 mean *mse* trends are analyzed- it becomes apparent that for all methods except *Kmeans withTarget*, aiming for a finer clustering by increasing c in general reduces mean *Normalized Median Error Increase* with strong negative correlations as measured by Pearson correlation coefficient $|\rho| > 0.5$, while increasing r^{train} increases $\overline{mse}_{r^{\text{train}}}$ with weak positive correlations $|\rho| < 0.5$. As *mse* measures relative median test-RMSE of *Training Sets* selected by method M to those resulting from randomly selected *Training Sets* of same (average) size, it can increase although the absolute median error resulting from method M decreases with increasing r^{train}.

The mean $\overline{mse}$ of all these methods, especially those using *withTarget*, indicates an only moderate improvement of around 9 to 14% in median. In the following we analyze results for specific hyperparameter values as those can differ strongly from $\overline{mse}$. We leverage the negative correlation between $\overline{mse}_c$ and c by regarding $c_{\max}$ only and choose $r^{\text{train}} = 0.1$ and $r^{\text{train}} = 0.2$ as representative train ratios whose smoothed random selection IQR, in the former case (for $n_{\text{train}} = 38$), just lays inside the *Work Set sSISSO* test-RMSE and standard deviation range and, in the latter case (for $n_{\text{train}} = 76$), whose 25% quartile for random selection lays quite close to *Work Set sSISSO* test-RMSE, and whose 75% quartile lays way below standard deviation. Results are displayed in table 4.3- the improvement in median error is about 8 to 40% for all methods and all IQR (displayed in brackets) lay below its random selection counterpart.

Table 4.3: *Normalized Median Error Increase* and *nIQR* (latter one in brackets) for $c = c_{\max}$ for different clustering methods in *PrimarySpace*.

r^{train}	*Kmeans noT*	*Kmeans withT*	*HDBSCAN noT*	*HDBSCAN withT*
0.1	0.617 (0.441)	0.928 (0.919)	0.887 (0.655)	0.708 (0.606)
0.2	0.727 (0.535)	0.806 (0.772)	0.872 (0.824)	0.748 (0.680)

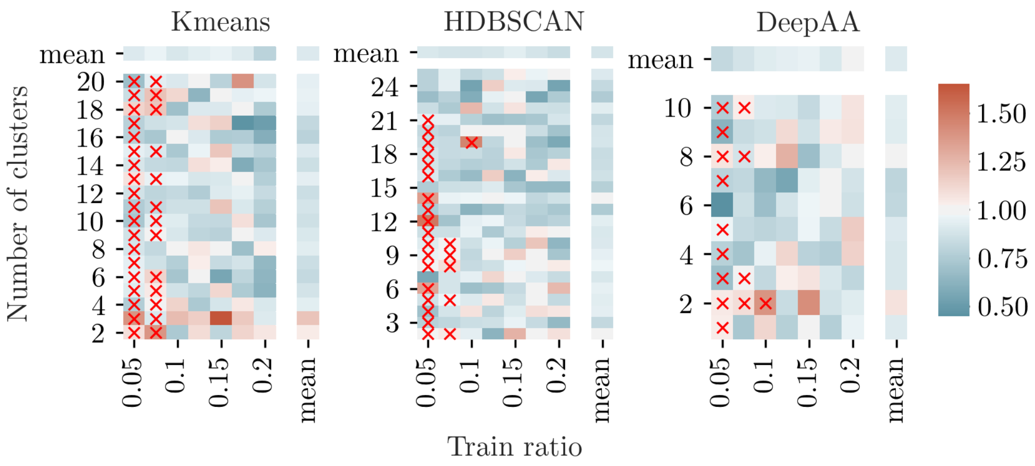


Figure 4.15: *Normalized Median Error Increase* (*nmei*) for *Kmeans* and *HDBSCAN* clustering in *DerivedSpace* and for clustering in *DeepAA*'s *Latent Space* for varying number of clusters c and train ratios r^{train} as well as their means for fixed c or r^{train}. Red crosses indicate that the cell's corresponding median test error exceeds the standard deviation of the target property.

4.2.4 Random Selection - Clustering in Derived Space

Results for clustering in *DerivedSpace* are displayed in fig. 4.15. Mean *Normalized Median Error Increases* are $\overline{mnei}^{\text{Kmeans}} = 0.900$, $\overline{mnei}^{\text{HDBSCAN}} = 0.874$ and $\overline{mnei}^{\text{DeepAA}} = 0.908$, again indicating only a small improvement in median of 9 to 13%. This time, mean *Normalized Median Error Increases* exhibit a weak negative correlation with c for all methods $|\rho| \in [0.22, 0.45]$. With r^{train}, a strong positive correlation results for *DeepAA* ($\rho = 0.8$) and a strong negative one for *Kmeans* and *HDBSCAN*, $\rho \in [-0.58, -0.53]$ (see fig. 4.16). As no clear trends can be extracted from these correlations regarding which hyperparameter areas yield best results, a further assessment of *nmei* and *nIQR* for a specific set of promising hyperparameters cannot be conducted.

4.2.5 Assessment of Best Equivalent Reduced Data Set Methods

In the following, we define which of our results presented in this section can be considered to be the "best", and corresponding training materials are compared in order to assess whether unique or "stable" *Representatives* exist for *Simple Cubic*

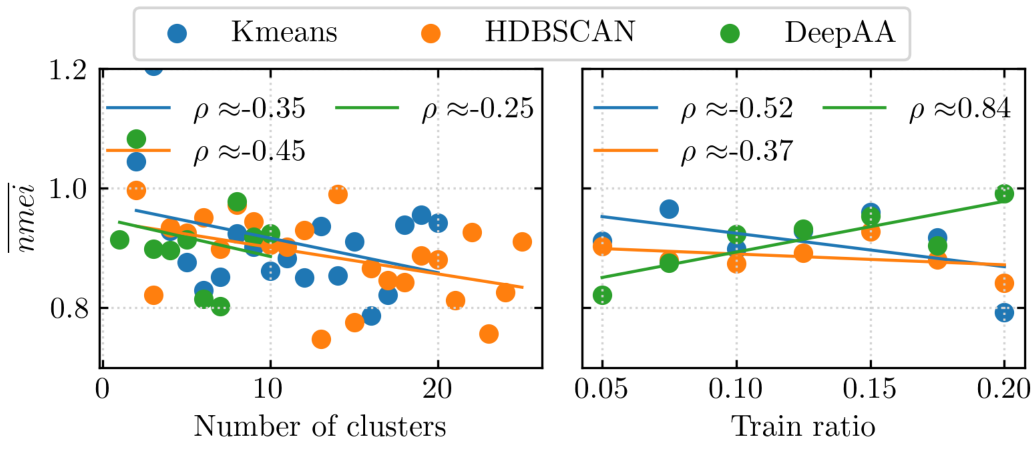


Figure 4.16: *Normalized Median Error Increase* (*nmei*) for *Kmeans* and *HDBSCAN* in *DerivedSpace* for *noTarget* and *withTarget* for varying number of clusters c and train ratios r^{train} as well as their means for fixed c or r^{train}. Red crosses indicate that the cell's corresponding median test error exceeds the standard deviation of the target property.

Perovskites with *Standard Test Set*.

As the negative correlation of *nmei* with number or ratio of training points, n_{train} or r^{train}, respectively, which appears in element metadata based clustering, is not sustained for clustering in *Primary* and *DerivedSpace*, and as we look for a substantially reduced data set in order to be able to call the constituting points *Representatives*, in the following we regard $r^{\text{train}} = 10\%$ and corresponding $n_{\text{train}} \in [30, 40]$ only. Furthermore, as a strong negative correlation of $\overline{nmei}$ with number of clusters c is only visible for clustering in *PrimarySpace*, hyperparameter c cannot be used to guide our choice of best models. We thus use an alternative criterion on which to base model choice, namely 3d-RMSE of *Validation Set* V, which is given by $V = W \setminus T$, where W is the *Work* and T the *Training Set*. Validation-RMSE e^{Val} is used instead of test-RMSE, so as to avoid introducing overfitting to the *Standard Test Set* with this final model selection step.

Two approaches for best model selection for $r^{\text{train}} = 10\%$ are implemented: In version (a), the $N = 16$ models of best e^{Val} for each of the 16 *Selection Methods* implemented in this chapter are chosen, the latter being *Guided Selection* for *ElA/B*, *GroupA/B* and *TypeA/B* clustering; *Cluster Center Selection* for *Kmeans*; *Random Selection* for *Kmeans* and *HDBSCAN* clustering- all conducted in *PrimarySpace* for the *noTarget* and *withTarget* version and in *DerivedSpace*; and *Random Selection* for clustering in *DeepAA*'s *Latent Space*. In version (b) all models that yield a $e^{\text{Val}} < 0.09$ are chosen, resulting in $N = 73$ selected models.

Comparing the $r^{\text{train}} = 10\%$ *Training Sets* of the so defined best models in detail is ineffective for our search for unique *Representatives* or *Exceptional Data Points*, which have to be included into training in order to extract a high quality model. Instead, for each material it is counted how often it is included in these *Training Sets*, in order to assess whether some materials are chosen (much) more frequently

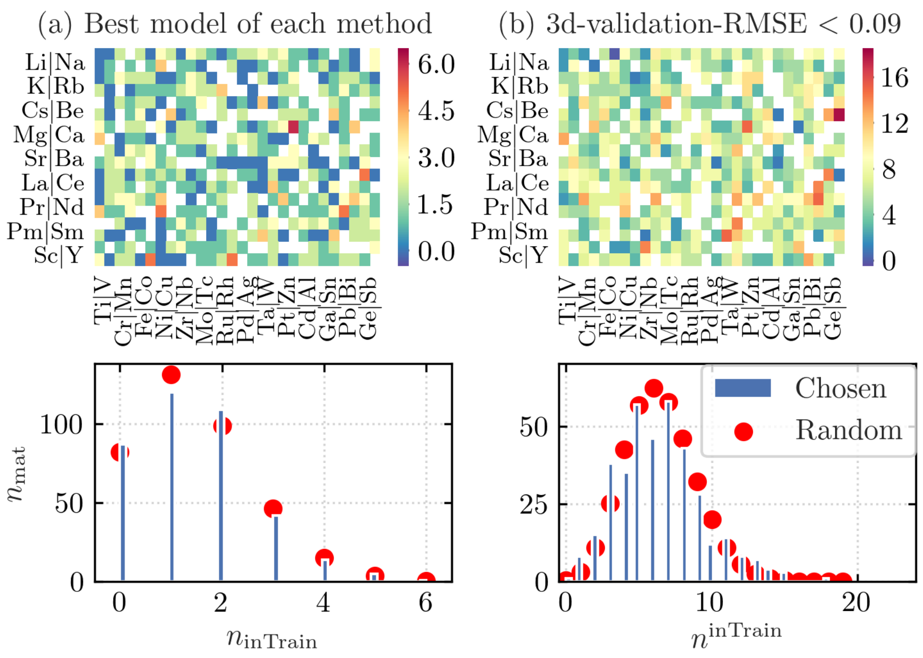


Figure 4.17: Assessment of whether unique *Representatives* can be found by comparing all *Training Set* materials of the best model in terms of 3d-validation-RMSE e^{Val} of each method, yielding 16 models in total, (left) and those of all models with $e^{\text{Val}} < 0.09$, hence in total 73 models, (right). In the illustrations of the top row, the color of each material indicates the number of times n_{inTrain} it is selected into the respective best-model *Training Set*. In the bottom row, histograms show the number of materials n_{mat} that have been selected n_{inTrain} times into the *Training Set*, along with the binomial distribution (red dots) that would result if the materials were picked completely randomly for each model, while keeping their number fixed. The distributions resemble one another closely.

than random. The results are shown in fig. 4.17.

At the top, heatmaps indicate the number of times a material has been selected into a *Training Set* with approach (a) and (b), respectively. As in fig. 4.8, element A is indicated by the vertical, element B by the horizontal axis. The first element name always represents the top row/ left column of its two adjacent rows/ columns, the second one the bottom/ right one. The colors are uniformly scattered; contrary to fig. 4.8, no clear shared patterns emerge, as for instance elements or groups that are over- or underrepresented.

The bottom row confirms these findings: Histograms show how many materials n_{mat} have been selected n_{inTrain} times into the *Training Sets* yielded by approaches (a) and (b) (blue bars). They are compared to the results of the binomial probability mass function

Table 4.4: List of derived features d_i for $i \in [1, 2, 3]$ that appear more than once for the $N = 16$ best models of each method in case (a), or whose ratio of appearance $r^a > 0.05$ in the $N = 73$ models of $e^{\mathrm{Val}} < 0.09$ in case (b).

(a) Best model of each method

r^a	d
0.375	$\exp(r_{sB}^3)$
0.25	$Z_B/(r_{pB}(IP_B + HO_B))$
0.125	$r_{sB}^3 \cdot \exp(EA_B)$
0.125	$(HO_B \cdot Z_B)/(IP_B \cdot r_{pB})$

(b) 3d-validation-RMSE < 0.09

r^a	d
0.438	$\exp(r_{sB}^3)$
0.192	$Z_B/(r_{pB}(IP_B + HO_B))$
0.151	$(HO_B \cdot Z_B)/(IP_B \cdot r_{pB})$
0.137	$(Z_B \cdot r_{sA})/(EN_B \cdot r_{pB})$
0.082	$r_{sB}^3 \cdot \exp(EA_B)$
0.068	$\exp(EA_B) \cdot \log(r_{sB})$
0.055	$(EN_B + HO_B)/(EA_B \cdot LU_B)$

$$n_{\mathrm{mat}}(n_{\mathrm{inTrain}}) \approx \binom{N}{n_{\mathrm{inTrain}}} p^{\,n_{\mathrm{inTrain}}} \,(1-p)^{N-n_{\mathrm{inTrain}}} \quad , \tag{4.5}$$

where $p = \frac{\overline{|T|}}{|W|}$ indicates the probability that any material inside the *Work Set* of size $|W|$ is selected randomly into a *Training Set* of average size $\overline{|T|}$. The number of regarded *Training Sets* is denoted N. Parameter values are $\overline{|T|} = 34.4$, $|W| = 378$, and $N = 16$ for approach (a) and $\overline{|T|} = 33.4$, $|W| = 378$, and $N = 73$ for approach (b). The results of this function are indicated by red dots and would appear if instead of choosing the *Training Sets* yielding the best models, each time the same number of materials was chosen randomly. They closely resemble the distributions yielded by best model selection (blue bars), indicating that no unique or stable set of *Representatives* exists, nor single materials which are more or less suitable as *Representatives*. Hence, for *Prototypes*, no *Exceptional Data Points* can be found.

In the following, our two model selection approaches are used in order to assess how stable or unique the derived features of the chosen best *sSISSO* models are. After that, a comparison of method performance, given the substantial data set reduction to 10%, is based on approach (b).

In table 4.4, derived features that appear for approach (a) and (b) are displayed alongside their ratio of appearance r^a. In case (a), only derived features which appear more than once for the $N = 16$ models are displayed, in case (b) only those chosen in more than 5% of the $N = 73$ models. Despite the non-stable *Representatives*, some derived features are repeatedly chosen, as $\exp(r_{sB}^3)$ with $r^a = 0.4$ in both cases. Feature $r_{sB}^3 \cdot \exp(EA_B)$ is the only one which also appears in *3d-sSISSO* when trained on the whole *Work Set*. Hence, although a variety of *Training Set* materials prove to be suitable *Representatives*, the best analytical models resulting from training on them do not exhibit a variety of same degree, as their derived features partly repeat.

An examination of the *Ratio of Models with Low Validation Error* r^*, thus of models with $e^{\mathrm{Val}} < 0.09$, reveals that *Kmeans Center Selection* in *DerivedSpace* is most successful, with $r^* = 0.2$, followed by *Kmeans Center Selection* in *noTarget PrimarySpace* and *Guided Random Subset Selection* for *GroupA* clustering, which both exhibit $r^* = 0.133$, and *Guided Random Subset Selection* for *TypeA* clustering with $r^* = 0.1$. All other training point selection methods exhibit an $r^* \in [0, 0.04]$. If substantial *Training Set* reductions to 10% are to be conducted, these best methods should preferred over those most successful in terms of average *nmei*'s.

5 Iterative Approach

As the last chapter proved that the set of *Prototypes*, thus *Average Representatives* of a data set is not unique, the *Iterative Approach* presented in this chapter aims to find a "stable" set of *Archetypes*, thus *Extreme Representatives* to answer the following question:

How can we estimate whether a new material of unknown target property is likely to be predicted well by our current best analytical model?

To this end, we aim to find *Extreme Representatives* or *Archetypes*, thus a subset of data points on which training *sSISSO* yields a model of comparable quality as training on the *Work Set*, which additionally are to exhibit a *Property* that can be used to make limiting statements about the prediction quality of materials of unknown target property.

The algorithm we designed to discover *Archetypes*, *Iterative Approach*, is presented in the following, alongside its two versions, followed by a guide on how results can be interpreted under which conditions, and some details on execution.

The algorithm works as follows: Initially, a random *Training Set* T_0 of given size n^T is selected from the *Work Set* W. Then, at each iteration step $i = \{0, 1, 2, ..., 200\}$, *sSISSO* of maximum dimension $d_{\max} = 3$ and *rung* $= 2$ is trained on it. The model corresponding to the dimension $d^* = \arg\min_d(e_d^{V_i})$ of smallest validation error $e_d^{V_i}$, with *Validation Set* $V_i = W \backslash T_i$, is chosen and a *Selection Method* which chooses the next *Training Set* T_{i+1} of same size is applied to it. As *SISSO* and all *Training Set Selection Methods* are deterministic, the iteration is stopped once the same *Training Set* is selected twice. If the reselection occurs in the subsequent step, $T_i = T_{i+1}$, we call the algorithm *converged*, if it occurs for any other future step, $T_i = T_{i+t}$ for $t > 1$, we say it reached a *loop*. Should none of those be reachable within $i_{\max} = 200$ iterations, we call the run *failed*.

Two different *Selection Methods* are used:

The first is the *Biggest Error Method* which, given the selected model, calculates the prediction error of each *Work Set* material, and exchanges up to a number of materials n^S from the *Training Set* with *Validation Set* materials, corresponding to a given *Maximum Swap Ratio* $r^S = n^S/n^T$. The exchange is conducted as follows: Remove the n^S materials of smallest prediction error from the *Training Set* T_i. Together with the n^S materials of biggest prediction error from the *Validation Set*, they form the *Potential Swap Set* of size $2n^S$. Select the n^S materials of biggest prediction error from it and add them back to the *Training Set*, obtaining T_{i+1}.

For $r^S = 1$, the *Biggest Error Selection Method* thus finds *Archetypes* $A_i = T_i$ that are worst described by model M_i as compared to all *Work Set* materials, thus setting an upper limit for prediction errors. If the algorithm *converges*, $A_i = A_{i+1} \Rightarrow M_i = M_{i+1}$, hence, M_i is also the best analytical model that can be extracted from *sSISSO* training on them. For $r^S = 1$, the entire *Training Set* can be exchanged at each iteration step. Due to that, both chosen dimension d^* and derived features of the model, d_i for $i \in \{1, ..., d^*\}$ are likely to change drastically from step to step, which might impair *convergence*- data points with *Biggest Error* in one model are likely to not still exhibit the *Biggest Error* if the model changes completely. For that, we also run the algorithm with $r^S < 1$ in order to facilitate *convergence*- if only few *Training Points* are exchanged, the change in optimal descriptor space is more likely to be more marginal as well. At the same time, for $r^S < 1$ the algorithm is more likely to get stuck in "local minima", meaning that *convergence* is reached although a large part of the *Training Set* points are not *Archetypes* in the sense of exhibiting *Biggest Error* for the chosen model. They are only kept as the *Maximum Swap Ratio* r^S did not allow for more exchanges in *Training Set* materials.

The second *Selection Method* is the *Linear Archetypes Method* as described in section 3.1.2- given the selected model and its descriptor space spanned by derived features d_i for $i \in \{1, ..., d^*\}$, it calculates the virtual linear *Archetypes* as the materials forming the convex hull of all *Work Set* materials and finds their actual-material counterparts as their nearest neighbors. This *Selection Method* depends on two hyperparameters: One is the number of Archetypes, which corresponds to the *Training Set* size $|A_i| = |T_i| = n^T$, the other is the *Relaxation Parameter* $\delta \in [0, 0.5]$, whose increase produces an outward shift of positions of virtual- and consequently also actual- *Archetypes* as illustrated in fig. 3.1. Irrespective of the chosen hyperparameters, upon *convergence*, *Archetypes* are found. Their *Property* is that, when *sSISSO* is trained onto them, it yields a descriptor space such that all *Work Set* materials are enclosed by the convex hull formed by these *Archetypes.* This convex hull hence defines a subspace in descriptor space for which we have relatively dense observations and for which the quality of predicting the target property of an unknown material is thus probably high. For a small $\delta \approx 0$, the convex hull does not enclose all *Work Set* materials (see fig. 3.1) and yields a conservative subspace of assumed high prediction quality. For $\delta \approx 0.5$ the opposite holds.

As expounded above, in order for the materials in the resulting *Training Set* to be considered *Archetypes*, with the *Selection Method Specific Property* of either representing the *Work Set* but at the same time exhibiting the largest prediction errors, or of defining a descriptor subspace of probably high prediction quality of unknown materials, as for *Biggest Error* and *Linear Archetypes Selection*, respectively, the algorithm must reach *convergence.*

If instead, a *loop* is reached, models and corresponding *Representatives* are yielded which do not exhibit the respective *Selection Method Specific Property*, as $M_{i-1} = M^{\mathrm{Prop}} \neq M_i = M^{\mathrm{Repr}} = M^{\mathrm{chosen}}$ and thus cannot be denoted *Archetypes* in the narrower sense. Here, M^{Prop} denotes the model in which the finally selected *Archetype* candidates exhibit their *Selection Method Specific Property.* Training on them results in a model M^{Repr} for the *Archetypes* candidates, which hopefully is *representative* for the data set, but in which the *Archetype* candidates no longer exhibit the desired *Selection Method Specific Property.* Hence, the only use for material subsets found by *Iterative Approach* ending in a *loop* would be as *Representatives.*

Analogously to *Equivalent Reduced Data Set* section of the previous chapter, resulting models are evaluated using *Normalized Median Error Increase nmei* (see eq. 4.3). It defines the ratio of increase of median test-RMSE resulting from choosing a *Training Set* with a given method, where training on the whole *Work Set* is the benchmark, to the smoothed median increase for a randomly selected *Training Set* of same size. The median test-RMSE of method M and *Training Set* size n, denoted $m^{M,n}(e)$ in eq. 4.3, are here defined as median test-RMSE of all models contained in the *loop*, respectively as test-RMSE of the *converged* model. In order for a yielded model or set of models to be considered valid and corresponding *Archetypes* or *Representative* candidates to be accepted as such, *nmei* < 1 must hold. Only then their definition as *(Extreme) Representatives*, upon which training yields a model of similar quality as compared to training on the whole *Work Set*, is met.

Due to the computational expensiveness of the *Iterative Approach*, in the following it is run once for each *Selection Method* and set of associated hyperparameters and assessed whether it *converges* and, if so, how the *Normalized Median Error Increase* depends on the hyperparameters. Please note that due to a fixed seed for the first random *Training Set* T_0 selection, all runs of same *Training Ratio* r^T start with the same *Training Set* $T_0(M, r^T, h) \equiv T_0(r^T)$, irrespective of *Selection Method* M and its associated hyperparameter h value, with $h \in [r^S, \delta]$.

5.1 Biggest Error Selection Method

In table 5.1, the convergence status and *Normalized Median Error Increase* for applying *Iterative Approach* with *Biggest Error Selection* to *Simple Cubic Perovskites* with target property *lattice constant* (see chapter 2) is displayed. Floats indicate the resulting *Normalized Median Error Increase.* Integer values in brackets signal a *loop*, denoting its loop length, *convergence* is marked by ✓, and *failure* by ✗. For adjacent cells of same result, the latter is printed once and its range of validity is denoted by adjoining colons.

It can be seen that only the run of $r^T = 0.6$ and $r^S = 0.3$ converges, with a very high *Normalized Median Error Increase nmei* $= 13.8$, indicating that the learned model describes the *Test Set* materials significantly worse than the model of median error resulting from randomly selected *Training Sets* of same size. The materials constituting the *Training Set* which results in the *converged* model thus do not fulfill the criteria to be considered *Archetypes.* As for all hyperparameter sets with *loop* results, corresponding *Normalized Median Error Increases nmei* ≥ 2.0 holds, yielding corresponding *Training Sets* useless as *Representatives.*

Considering the high computational cost of running this algorithm, above results of one run with each hyperparameter set are not promising enough to justify further investigations for the *Iterative Approach* with *Biggest Error Selection*, just as with *Linear Archetypes Selection*, as is expounded in the next section.

5.2 Linear Archetypes Selection Method

For none of the hyperparameter sets, *convergence* is achieved, and almost all *loop* solutions exhibit a *Normalized Median Error Increase nmei* > 1, rendering them unfit to extract *Representatives* from them.

An exception are the two hyperparameter sets $(r^T, \delta) \in \{(0.9, 0.1), (0.9, 0.5)\}$.

Table 5.1: *Normalized Median Error Increase (Convergence status)* for applying *Iterative Approach* with *Biggest Error Selection* to *Simple Cubic Perovskites* with target property *lattice constant* (see chapter 2). Integer values in brackets indicate that the run ended in a *loop*, denoting its loop length, *convergence* is marked by ✓, *failure* by ✗. For runs of same *Training Ratio* r^T and differing *Maximum Swap Ratio* r^S with same result, the latter is printed once and its range of validity is marked by adjacent colons. For *nmei* < 1, the (set of) *Representative* candidates are accepted as such; if additionally *convergence* is reached, they are considered *Archetypes* with *Property* of setting an upper limit for prediction errors.

		Training Ratio r^T								
		0.1	0.2	0.3	0.4	0.5	0.6	0.7	0.8	0.9
Maximum Swap Ratio r^S	0.1	2.7 (4)	✗	✗	✗	✗	10.2 (3)	4.4 (3)	✗	:
	0.2	✗	5.8 (2)	✗	✗	✗	18.1 (2)	✗	:	:
	0.3	✗	5.9 (6)	✗	✗	✗	13.8 ✓	:	:	:
	0.4	2.7 (2)	✗	✗	7.1 (162)	✗	:	:	:	:
	0.5	2.0 (81)	✗	✗	✗	✗	:	:	5.9 (114)	22.1 (2)
	0.6	2.0 (95)	5.9 (6)	✗	✗	✗	11.9 (67)	14.7 (72)	:	:
	0.7	✗	✗	✗	✗	✗	:	:	:	:
	0.8	✗	5.8 (2)	✗	✗	✗	:	:	:	:
	0.9	✗	5.8 (2)	✗	✗	✗	:	:	:	:

Their *nmei* $= -46.6$ and *nmei* $= -46.0$, respectively, are negative due to a smoothed median error of randomly selected *Training Set* to *Work Set* trained error ratio $m^s(e^{r,0.9 \cdot n^T})/e^{\text{Work}} = 0.999 < 1$, which leads to a negative denominator of eq. 4.3, entailing that median *loop* model error $m(e^{M,n^T}) > e^{\text{Work}}$. Corresponding *Training Sets* are thus not usable as *Representatives*.

Applying this insight to the other, positive *nmei loop* solutions for *Linear Archetype Selection* and $(r^T, \delta) = (0.9, 0.4)$, and -in previous section- *Biggest Error Selection* with $r^T = 0.9$ and $r^S \in \{0.1, 0.2, ..., 0.9\}$, yielding *nmei* $= 23.7$ and *nmei* $= 22.1$, respectively, implies that their median error of *loop* models is 2.37%, respectively 2.21% lower than e^{Work}. As this improvement is marginal considering the large *Training Ratio* $r^T = 0.9$, the corresponding *Training Sets* are not useful as *Representatives*.

In summary, analogously to the previous section, due to the high computational cost of the *Iterative Approach*, these one-time run results for *Linear Archetype Selection* are not promising, and the *Iterative Approach* with any *Selection Method* is not investigated further.

5.3 Assessment of Iterative Approach

The *Iterative Approach* did not produce promising results for any *Selection Method* and hyperparameter set. In the following, possible reasons for that are discussed along with suggestions for enhanced future design of *Representative* detection algo-

Table 5.2: *Normalized Median Error Increase (Convergence status)* for applying *Iterative Approach* with *Linear Archetype Selection* to *Simple Cubic Perovskites* with target property *lattice constant* (see chapter 2). Integer values in brackets indicate that the run ended in a *loop*, denoting its loop length, *convergence* is marked by ✓, *failure* by ✗. For *nmei* < 1, the (set of) *Representative* candidates are accepted as such; if additionally *convergence* is reached, they are considered *Archetypes* with *Property* of defining a descriptor subspace of probably high prediction quality.

		Relaxation Parameter δ				
		0.1	0.2	0.3	0.4	0.5
Training Ratio r^T	0.1	2.2 (2)	1.9 (6)	1.1 (9)	2.1 (4)	2.2 (2)
	0.2	2.1 (3)	2.7 (49)	3.1 (7)	2.5 (79)	3.1 (3)
	0.3	1.7 (11)	2.0 (80)	✗	1.8 (75)	1.8 (31)
	0.4	3.2 (9)	2.4 (59)	✗	✗	✗
	0.5	✗	3.5 (15)	3.1 (99)	2.6 (32)	2.9 (181)
	0.6	6.5 (10)	✗	2.7 (35)	✗	✗
	0.7	8.0 (21)	6.4 (77)	✗	7.6 (50)	✗
	0.8	✗	✗	17.9 (13)	✗	✗
	0.9	-46.6 (38)	✗	✗	23.7 (65)	-46.0 (100)

rithms.

- **No mechanism to ensure convergence nor systematic error decrease:** The algorithm is designed to find *Extreme Representatives* if it *converges*, hence, if inside the space spanned by the *sSISSO* model after training on *Archetype* candidates, the latter are selected again by the given *Selection Method.* The resulting model, which after *convergence* is always the same, additionally is to exhibit a reasonably small *sSISSO* test error.

  In order to achieve the latter goal, other ML algorithms are mathematically designed to systematically decrease the validation error defined by a cost function, possibly after a transient initial period. For instance, *DeepAA* optimizes its objective function (see eq. 3.2) for successful image interpolation. Clearly, the *sSISSO* application at each step i likewise uses optimization of its cost function (see eq. 3.3) to find the best model for given *Training Set* materials. The combination of *sSISSO* and any of the *Selection Methods* to our *Iterative Approach* on the other hand lacks mechanisms which ensures systematic error decrease and *convergence.* Introducing such a mechanism to *Iterative Approach* in its current form seems difficult as the iterative reapplication of *sSISSO* allows for the model and thus basis of error calculation and *Representative* candidates selection to drastically change.

- **Simultaneous pursuit of conflicting goals:** For *Biggest Error Selection Method* with $r^S = 1$, we hope to find a *Training Set* whose *sSISSO* model predicts the target properties of all *Training Set* materials worse than the ones of all other *Work Set* materials, in order to exhibit *Biggest Error Property.*

That very model though is selected by *sSISSO* minimizing training error, thus the average error of *Training Set* materials, irrespective of the average error of the other *Work Set* materials, the validation error, which is hence prone to be larger. Regarding training and validation error of all models analyzed in this thesis (both resulting from *Direct* and *Iterative Approach*), the former is larger than the latter in less than two percent of the cases. Please note that a training error larger than validation error is a necessary, but not sufficient condition for *Biggest Error Property* with $r^S = 1$, which poses a much tighter constraint. The model selection mechanism of *sSISSO* thus conflicts with *Biggest Error Property* for large r^S, although in very rare cases, achieving both might be possible. In this work, *convergence* was only accomplished once and only for a small $r^S = 0.3$.

- **Unknown validity of *Archetype Property* implications:** For *Biggest Error Selection*, the *Property* of training errors exceeding validation errors is to be achieved, and assumed to imply an upper limit for prediction errors of other unknown materials as well; for *Linear Archetype Selection*, the desired *Property* is that the *Training Set* forms a convex hull around the *Work Set*, which is presumed to define a subspace inside which new material predictions are valid, while outside of it no statements can be made, especially for material positions far away from the hull. It has neither been proved nor empirically investigated though whether these two implications are valid. A possible explanation for training errors as an upper limit for validation errors could be mere statistical chance instead of a reliable underlying mechanism which also applies to other unknown, similar materials. The same holds for *Linear Archetype Selection*: Whether the position of an unknown data point inside or outside of the convex hull spanned by the *Training Set* in general is a suitable predictor for its estimation quality is unclear. Intuition guided by interpolation and extrapolation does provide a reason for this approach; yet, an empirical verification of its applicability to models generated by *sSISSO* prior to *Iterative Approach* design and implementation should have been conducted.

To prevent the above errors for future designs of iterative algorithms of this kind, following procedure is suggested: First, a clear single goal or question is to be formulated, like *How can we estimate whether a new material of unknown target property is likely to be predicted well by our current best analytical model?*, which is then broken down into its implicit sub goals, here being *Find the best analytical model for a given* Work Set *or one of similar quality* and *Find a prediction success probability metric for unknown data points.* For each sub goal, different approaches can arise. An approach should only be considered valid if its algorithm systematically decreases the cost function of the respective sub goal- if it does, its performance should be tested empirically if possible. All combinations of valid and performative sub goal approaches should be checked for compatibility, and the performance of the remaining combinations should finally be evaluated empirically.

6 Conclusion

The goal of this work is to derive methods for discovering underlying structures in a given data set, which can be used to enhance the prediction of a target property and to identify representative data points for efficient training. Such *Representatives* should yield a model of similar quality as training on the whole data set, and in one version of them, *Prototypes*, they additionally are to mirror the underlying cluster structure, while in the other version, *Archetypes*, they are to allow for limiting statements about the quality of future unknown material predictions. We also investigate whether *Exceptional Data Points* exist, thus points that are indispensable in a set of *Representatives*. Using the *Cubic Perovskites* data set, which exhibits reliable data quality and is sufficiently large, and method evaluation metrics based on median test errors, which ensures statistical significance of our findings, we approach this goal by answering three core questions:

Does a given data set have a structure, or subsets of materials that follow different laws than others? To answer this question, we evaluate the quality of structures returned by different clustering methods with the median test error of *mSISSO*, which allows for differing fitting coefficients for each cluster.

Some of the investigated clustering methods are based on chemical insight about *Simple Cubic Perovskites* ABO_3, whose A elements all belong to group one to three and the f-block of the periodic table, and whose B elements belong to group four and above. Their comparative assessment shows that *ElB* clustering, which assigns all materials of same B element to one cluster, and *GroupB* clustering, that allots materials to one cluster if their B element belongs to the same group in the periodic table, are by far most successful. Their reduction in median test error amounts to more than 67% as compared to a global *sSISSO* application, which yields a best model of fitting coefficients that are globally valid for all materials. In order to apply these approaches to generic data sets, an adaptation that allows for overlapping clustering is presented whose performance remains to be tested in future work.

Other methods extract prototypical clusters in primary feature space (*PrimaryS-pace*), which either includes or excludes target property (*withTarget* or *noTarget*), or in the space derived by *sSISSO* (*DerivedSpace*). Of those, *Kmeans*, which returns isotropic clusters of equal variance, which thus represent uniform feature space partitions, and *HDBSCAN*, which does not share these constraints and hence reflects the underlying cluster structure much more closely, prove to be successful in *noTarget PrimarySpace* and in *DerivedSpace* with a *Number of Clusters* $c \in [5, 8]$. For them, error reduction amounts to 30 to 35%, and they can be applied to generic data sets without adaptations.

All of the above mentioned successful clustering methods show satisfying results globally and locally, and their resulting cluster structures are partly similar. It can be shown that the degree to which clusters in the given data are clear-cut, and resulting unambiguity of partition with the stochastic algorithm *Kmeans*, is an indicator for the performance of the latter.

We can thus conclude that the presented methods are suitable to find a meaningful data set structure. The material subsets defined by the latter each follow slightly different laws, in the sense that allowing for different *mSISSO* coefficient fits for each of them substantially reduces median test error.

Based on such structures, in a next step we aim to answer the following:

Can the data set be reduced substantially, such that training the model still yields results of similar quality, and are there stable or unique data points whose inclusion during training is strictly necessary in order to obtain such a model?

For the first part of this question, we use the success measure *Normalized Median Error Increase* (*nmei*) which is designed to ensure statistical significance of our findings and which should be as small as possible.

A global comparison shows that most successful clustering methods are again based on chemical insight and a *Guided Random Selection* from resulting clusters. *ElA* clustering, which allots all materials of same element *A* to one cluster, yields by far best results, with a mean median test error even lower than the one resulting from *sSISSO* on the non-reduced data set. Next most successful methods are *GroupA* clustering, which assigns materials whose element *A* belongs to the same group in periodic table, *ElB*, and *TypeA* clustering, which yields four clusters of materials whose element *A* is either an *Alkali Metal*, *Alkaline Earth Metal*, *Lanthanide*, or *Transition Metal*. Their mean *nmei*s range between 31 and 66% - it has to be kept in mind though, that the performance of the proposed adaptation of these methods to generic data sets still remains to be tested.

Next most fruitful methods prove to be those based on clustering in feature spaces, namely *Cluster Center Selection* based on *Kmeans* in *DerivedSpace*, which slightly outperform application in *noTarget PrimarySpace*, followed by *Random Selection* inside each cluster yielded by either *Kmeans* or *HDBSCAN* in *noTarget* or *withTarget PrimarySpace*, or in *DerivedSpace*. Their *nmei*'s range between 69 and 92%, and especially methods using *Random Selection* only marginally improve median error.

No reliable set of optimal hyperparameters can be found for these methods, as correlations between either of them and *nmei* prove to be inconsistent. A guided presetting of hyperparameters in case of a new application is thus not possible, and neither is choosing best models based on hyperparameters for a final comparative method evaluation.

The latter is thus conducted specifically for substantial data set reductions to 10%, using a method success metric based on a validation error threshold. In this assessment, best methods are *Cluster Center Selection* for *Kmeans* in *DerivedSpace*, which outperforms its counterpart in *noTarget PrimarySpace* and *Guided Random Selection* based on *GroupA* clustering, followed by *Guided Random Selection* based on *TypeA* clustering.

It can hence be concluded that reducing the data set is possible, and doing so in an informed manner, thus with the presented methods, yields results which are reliably better then for random data set reductions. If the data set reduction is to be substantial, methods excelling in terms of the latter success measure should be preferred over those with best global *nmei*.

Regarding the second part of the question, it can be shown that no unique or stable *Representatives* exist that have to be included in training in order to obtain a high quality model- it can thus be concluded for *Prototypes* that such *Exceptional Data Points* do not exist. This finding probably applies to general data sets- any material might be included or excluded as *Representative*, as long as the combination of *Representatives* is conducted in the right manner, for instance using the presented most successful clustering methods.

Irrespective of which success measure is chosen, almost none of the clustering methods which perform best for finding an *Equivalent Reduced Data Set* (which corresponds to our second question) do so as well for *Structure Exploitation* (which corresponds to our first question). The performance of a method regarding one of these two goals does thus not predict its performance in the other one.

The attempt of answering our third question,

How can we estimate whether a new material of unknown target property is likely to be predicted well by our current best analytical model?,

thus of discovering *Archetypes* with the *Iterative Approach* fails for both its *Biggest Error* and *Linear Archetypes* version. Probable causes for its failure are extracted and a procedure is suggested to prevent them in the future.

Bibliography

[1] Madeleine Udell. *Generalized low rank models*. Stanford University, 2015.

[2] Leland McInnes. A bluffer's guide to dimension reduction, 2018.

[3] Sebastian Mathias Keller, Maxim Samarin, Fabricio Arend Torres, Mario Wieser, and Volker Roth. Learning extremal representations with deep archetypal analysis. *arXiv preprint arXiv:2002.00815*, 2020.

[4] Giancarlo Ragozini, Francesco Palumbo, and Maria Rosaria D'Esposito. Archetypal analysis for data-driven prototype identification. *Statistical Analysis and Data Mining: The ASA Data Science Journal*, 10(1):6–20, 2017.

[5] David L Davies and Donald W Bouldin. A cluster separation measure. *IEEE transactions on pattern analysis and machine intelligence*, (2):224–227, 1979.

[6] Franz Aurenhammer. Voronoi diagrams—a survey of a fundamental geometric data structure. *ACM Computing Surveys (CSUR)*, 23(3):345–405, 1991.

[7] Leland McInnes, John Healy, and Steve Astels. hdbscan: Hierarchical density based clustering. *Journal of Open Source Software*, 2(11):205, 2017.

[8] Adele Cutler and Leo Breiman. Archetypal analysis. *Technometrics*, 36(4):338–347, 1994.

[9] Morten Mørup and Lars Kai Hansen. Archetypal analysis for machine learning and data mining. *Neurocomputing*, 80:54–63, 2012.

[10] Runhai Ouyang, Stefano Curtarolo, Emre Ahmetcik, Matthias Scheffler, and Luca M Ghiringhelli. SISSO: A compressed-sensing method for identifying the best low-dimensional descriptor in an immensity of offered candidates. *Physical Review Materials*, 2(8):083802, 2018.

[11] Runhai Ouyang, Emre Ahmetcik, Christian Carbogno, Matthias Scheffler, and Luca M Ghiringhelli. Simultaneous learning of several materials properties from incomplete databases with multi-task SISSO. *Journal of Physics: Materials*, 2(2):024002, 2019.

[12] Taffee T Tanimoto. Elementary mathematical theory of classification and prediction. 1958.

[13] Paul Jaccard. The distribution of the flora in the alpine zone. 1. *New phytologist*, 11(2):37–50, 1912.

[14] RGD Steel and JH Torrie. Principles and procedures of statistics; with special reference to the biological sciences. 1960.

Appendix

A Clusters yielded by Element Metadata based Clustering

A.1 EIA Clustering

Materials of same element A are assigned to one cluster.

- **Li - cluster:** $LiAgO_3$, $LiAlO_3$, $LiBiO_3$, $LiCdO_3$, $LiCoO_3$, $LiCrO_3$, $LiCuO_3$, $LiFeO_3$, $LiGaO_3$, $LiMoO_3$, $LiNbO_3$, $LiNiO_3$, $LiPbO_3$, $LiPdO_3$, $LiPtO_3$, $LiRhO_3$, $LiRuO_3$, $LiSbO_3$, $LiTaO_3$, $LiTcO_3$, $LiTiO_3$, $LiVO_3$, $LiWO_3$, $LiZnO_3$, $LiZrO_3$

- **Na - cluster:** $NaAgO_3$, $NaAlO_3$, $NaBiO_3$, $NaCoO_3$, $NaGaO_3$, $NaMnO_3$, $NaMoO_3$, $NaNbO_3$, $NaNiO_3$, $NaPbO_3$, $NaPdO_3$, $NaPtO_3$, $NaRhO_3$, $NaSbO_3$, $NaSnO_3$, $NaTaO_3$, $NaTiO_3$, $NaVO_3$, $NaZnO_3$, $NaZrO_3$

- **K - cluster:** $KBiO_3$, $KCoO_3$, $KCrO_3$, $KCuO_3$, $KFeO_3$, $KGaO_3$, $KGeO_3$, $KMnO_3$, $KMoO_3$, $KNbO_3$, $KNiO_3$, $KPbO_3$, $KPdO_3$, $KPtO_3$, $KSbO_3$, $KSnO_3$, $KTaO_3$, $KTcO_3$, $KTiO_3$, KVO_3, KWO_3, $KZnO_3$, $KZrO_3$

- **Rb - cluster:** $RbAgO_3$, $RbBiO_3$, $RbCoO_3$, $RbCuO_3$, $RbFeO_3$, $RbGaO_3$, $RbGeO_3$, $RbMnO_3$, $RbMoO_3$, $RbNbO_3$, $RbNiO_3$, $RbPbO_3$, $RbPtO_3$, $RbRuO_3$, $RbSbO_3$, $RbSnO_3$, $RbTcO_3$, $RbTiO_3$, $RbVO_3$, $RbZnO_3$, $RbZrO_3$

- **Cs - cluster:** $CsAgO_3$, $CsCdO_3$, $CsCoO_3$, $CsCrO_3$, $CsFeO_3$, $CsMnO_3$, $CsMoO_3$, $CsNbO_3$, $CsNiO_3$, $CsPbO_3$, $CsPdO_3$, $CsPtO_3$, $CsRhO_3$, $CsRuO_3$, $CsSbO_3$, $CsSnO_3$, $CsTaO_3$, $CsVO_3$, $CsWO_3$, $CsZrO_3$

- **Be - cluster:** $BeAlO_3$, $BeCdO_3$, $BeCoO_3$, $BeCuO_3$, $BeFeO_3$, $BeGeO_3$, $BeMnO_3$, $BeNbO_3$, $BePbO_3$, $BePdO_3$, $BePtO_3$, $BeRuO_3$, $BeSbO_3$, $BeSnO_3$, $BeTiO_3$, $BeVO_3$, $BeZnO_3$, $BeZrO_3$

- **Mg - cluster:** $MgAgO_3$, $MgBiO_3$, $MgCdO_3$, $MgCuO_3$, $MgFeO_3$, $MgMnO_3$, $MgMoO_3$, $MgNbO_3$, $MgNiO_3$, $MgPbO_3$, $MgPdO_3$, $MgPtO_3$, $MgSnO_3$, $MgTaO_3$, $MgTcO_3$, $MgTiO_3$, $MgVO_3$, $MgWO_3$, $MgZnO_3$, $MgZrO_3$

- **Ca - cluster:** $CaAgO_3$, $CaAlO_3$, $CaCdO_3$, $CaCoO_3$, $CaCrO_3$, $CaCuO_3$, $CaFeO_3$, $CaGaO_3$, $CaMnO_3$, $CaMoO_3$, $CaNbO_3$, $CaPbO_3$, $CaPdO_3$, $CaPtO_3$, $CaRhO_3$, $CaSbO_3$, $CaTaO_3$, $CaTiO_3$, $CaWO_3$, $CaZrO_3$

- **Sr - cluster:** $SrAgO_3$, $SrAlO_3$, $SrCoO_3$, $SrCrO_3$, $SrFeO_3$, $SrGeO_3$, $SrMnO_3$, $SrMoO_3$, $SrNbO_3$, $SrNiO_3$, $SrPbO_3$, $SrRhO_3$, $SrSbO_3$, $SrTaO_3$, $SrTcO_3$, $SrVO_3$, $SrWO_3$, $SrZnO_3$, $SrZrO_3$

- **Ba - cluster:** $BaAgO_3$, $BaAlO_3$, $BaBiO_3$, $BaCdO_3$, $BaCoO_3$, $BaCrO_3$, $BaCuO_3$, $BaGaO_3$, $BaGeO_3$, $BaMnO_3$, $BaNiO_3$, $BaPbO_3$, $BaPdO_3$, $BaPtO_3$, $BaRhO_3$, $BaRuO_3$, $BaTaO_3$, $BaTcO_3$, $BaTiO_3$, $BaVO_3$, $BaWO_3$, $BaZnO_3$

- **La - cluster:** $LaAgO_3$, $LaAlO_3$, $LaCoO_3$, $LaCrO_3$, $LaCuO_3$, $LaFeO_3$, $LaGaO_3$, $LaGeO_3$, $LaMnO_3$, $LaMoO_3$, $LaNbO_3$, $LaNiO_3$, $LaPdO_3$, $LaPtO_3$, $LaRuO_3$, $LaSbO_3$, $LaSnO_3$, $LaTaO_3$, $LaTiO_3$, $LaVO_3$, $LaWO_3$, $LaZnO_3$, $LaZrO_3$

- **Ce - cluster:** $CeAgO_3$, $CeBiO_3$, $CeCdO_3$, $CeCoO_3$, $CeCrO_3$, $CeGaO_3$, $CeGeO_3$, $CeMnO_3$, $CeMoO_3$, $CeNbO_3$, $CeNiO_3$, $CePtO_3$, $CeRhO_3$, $CeRuO_3$, $CeSnO_3$, $CeTaO_3$, $CeTcO_3$, $CeTiO_3$, $CeVO_3$, $CeWO_3$, $CeZrO_3$

- **Pr - cluster:** $PrAgO_3$, $PrAlO_3$, $PrBiO_3$, $PrCoO_3$, $PrCrO_3$, $PrFeO_3$, $PrGaO_3$, $PrGeO_3$, $PrMnO_3$, $PrNbO_3$, $PrNiO_3$, $PrPbO_3$, $PrPtO_3$, $PrRuO_3$, $PrSbO_3$, $PrSnO_3$, $PrTcO_3$, $PrTiO_3$, $PrVO_3$, $PrWO_3$, $PrZnO_3$

- **Nd - cluster:** $NdAgO_3$, $NdAlO_3$, $NdBiO_3$, $NdCdO_3$, $NdCrO_3$, $NdGaO_3$, $NdGeO_3$, $NdMnO_3$, $NdMoO_3$, $NdNbO_3$, $NdNiO_3$, $NdPbO_3$, $NdPtO_3$, $NdRuO_3$, $NdSbO_3$, $NdSnO_3$, $NdTaO_3$, $NdTiO_3$, $NdVO_3$, $NdZrO_3$

- **Pm - cluster:** $PmAlO_3$, $PmBiO_3$, $PmCdO_3$, $PmCrO_3$, $PmCuO_3$, $PmFeO_3$, $PmGaO_3$, $PmGeO_3$, $PmMnO_3$, $PmMoO_3$, $PmNbO_3$, $PmNiO_3$, $PmPbO_3$, $PmPdO_3$, $PmSbO_3$, $PmSnO_3$, $PmTaO_3$, $PmTcO_3$, $PmWO_3$, $PmZnO_3$, $PmZrO_3$

- **Sm - cluster:** $SmAgO_3$, $SmAlO_3$, $SmBiO_3$, $SmCdO_3$, $SmCrO_3$, $SmFeO_3$, $SmGaO_3$, $SmGeO_3$, $SmNbO_3$, $SmNiO_3$, $SmPtO_3$, $SmRhO_3$, $SmRuO_3$, $SmSbO_3$, $SmSnO_3$, $SmTaO_3$, $SmTcO_3$, $SmTiO_3$, $SmVO_3$, $SmWO_3$, $SmZrO_3$

- **Sc - cluster:** $ScAlO_3$, $ScCoO_3$, $ScCrO_3$, $ScCuO_3$, $ScGaO_3$, $ScMnO_3$, $ScMoO_3$, $ScNbO_3$, $ScNiO_3$, $ScPdO_3$, $ScRhO_3$, $ScRuO_3$, $ScSbO_3$, $ScSnO_3$, $ScTcO_3$, $ScTiO_3$, $ScWO_3$, $ScZrO_3$

- **Y - cluster:** $YAgO_3$, $YAlO_3$, $YBiO_3$, $YCdO_3$, $YCoO_3$, $YCrO_3$, $YCuO_3$, $YFeO_3$, $YGaO_3$, $YGeO_3$, $YMnO_3$, $YMoO_3$, $YNiO_3$, $YPbO_3$, $YPtO_3$, $YRhO_3$, $YRuO_3$, $YSnO_3$, $YTaO_3$, $YTcO_3$, $YTiO_3$, YVO_3, YWO_3, $YZnO_3$, $YZrO_3$

A.2 EIB Clustering

Materials of same element B are assigned to one cluster.

- **Ti - cluster:** $BaTiO_3$, $BeTiO_3$, $CaTiO_3$, $CeTiO_3$, $KTiO_3$, $LaTiO_3$, $LiTiO_3$, $MgTiO_3$, $NaTiO_3$, $NdTiO_3$, $PrTiO_3$, $RbTiO_3$, $ScTiO_3$, $SmTiO_3$, $YTiO_3$

- **V - cluster:** $BaVO_3$, $BeVO_3$, $CeVO_3$, $CsVO_3$, KVO_3, $LaVO_3$, $LiVO_3$, $MgVO_3$, $NaVO_3$, $NdVO_3$, $PrVO_3$, $RbVO_3$, $SmVO_3$, $SrVO_3$, YVO_3

- **Cr - cluster:** $BaCrO_3$, $CaCrO_3$, $CeCrO_3$, $CsCrO_3$, $KCrO_3$, $LaCrO_3$, $LiCrO_3$, $NdCrO_3$, $PmCrO_3$, $PrCrO_3$, $ScCrO_3$, $SmCrO_3$, $SrCrO_3$, $YCrO_3$

- **Mn - cluster:** $BaMnO_3$, $BeMnO_3$, $CaMnO_3$, $CeMnO_3$, $CsMnO_3$, $KMnO_3$, $LaMnO_3$, $MgMnO_3$, $NaMnO_3$, $NdMnO_3$, $PmMnO_3$, $PrMnO_3$, $RbMnO_3$, $ScMnO_3$, $SrMnO_3$, $YMnO_3$
- **Fe - cluster:** $BeFeO_3$, $CaFeO_3$, $CsFeO_3$, $KFeO_3$, $LaFeO_3$, $LiFeO_3$, $MgFeO_3$, $PmFeO_3$, $PrFeO_3$, $RbFeO_3$, $SmFeO_3$, $SrFeO_3$, $YFeO_3$
- **Co - cluster:** $BaCoO_3$, $BeCoO_3$, $CaCoO_3$, $CeCoO_3$, $CsCoO_3$, $KCoO_3$, $LaCoO_3$, $LiCoO_3$, $NaCoO_3$, $PrCoO_3$, $RbCoO_3$, $ScCoO_3$, $SrCoO_3$, $YCoO_3$
- **Ni - cluster:** $BaNiO_3$, $CeNiO_3$, $CsNiO_3$, $KNiO_3$, $LaNiO_3$, $LiNiO_3$, $MgNiO_3$, $NaNiO_3$, $NdNiO_3$, $PmNiO_3$, $PrNiO_3$, $RbNiO_3$, $ScNiO_3$, $SmNiO_3$, $SrNiO_3$, $YNiO_3$
- **Cu - cluster:** $BaCuO_3$, $BeCuO_3$, $CaCuO_3$, $KCuO_3$, $LaCuO_3$, $LiCuO_3$, $MgCuO_3$, $PmCuO_3$, $RbCuO_3$, $ScCuO_3$, $YCuO_3$
- **Zr - cluster:** $BeZrO_3$, $CaZrO_3$, $CeZrO_3$, $CsZrO_3$, $KZrO_3$, $LaZrO_3$, $LiZrO_3$, $MgZrO_3$, $NaZrO_3$, $NdZrO_3$, $PmZrO_3$, $RbZrO_3$, $ScZrO_3$, $SmZrO_3$, $SrZrO_3$, $YZrO_3$
- **Nb - cluster:** $BeNbO_3$, $CaNbO_3$, $CeNbO_3$, $CsNbO_3$, $KNbO_3$, $LaNbO_3$, $LiNbO_3$, $MgNbO_3$, $NaNbO_3$, $NdNbO_3$, $PmNbO_3$, $PrNbO_3$, $RbNbO_3$, $ScNbO_3$, $SmNbO_3$, $SrNbO_3$
- **Mo - cluster:** $CaMoO_3$, $CeMoO_3$, $CsMoO_3$, $KMoO_3$, $LaMoO_3$, $LiMoO_3$, $MgMoO_3$, $NaMoO_3$, $NdMoO_3$, $PmMoO_3$, $RbMoO_3$, $ScMoO_3$, $SrMoO_3$, $YMoO_3$
- **Tc - cluster:** $BaTcO_3$, $CeTcO_3$, $KTcO_3$, $LiTcO_3$, $MgTcO_3$, $PmTcO_3$, $PrTcO_3$, $RbTcO_3$, $ScTcO_3$, $SmTcO_3$, $SrTcO_3$, $YTcO_3$
- **Ru - cluster:** $BaRuO_3$, $BeRuO_3$, $CeRuO_3$, $CsRuO_3$, $LaRuO_3$, $LiRuO_3$, $NdRuO_3$, $PrRuO_3$, $RbRuO_3$, $ScRuO_3$, $SmRuO_3$, $YRuO_3$
- **Rh - cluster:** $BaRhO_3$, $CaRhO_3$, $CeRhO_3$, $CsRhO_3$, $LiRhO_3$, $NaRhO_3$, $ScRhO_3$, $SmRhO_3$, $SrRhO_3$, $YRhO_3$
- **Pd - cluster:** $BaPdO_3$, $BePdO_3$, $CaPdO_3$, $CsPdO_3$, $KPdO_3$, $LaPdO_3$, $LiPdO_3$, $MgPdO_3$, $NaPdO_3$, $PmPdO_3$, $ScPdO_3$
- **Ag - cluster:** $BaAgO_3$, $CaAgO_3$, $CeAgO_3$, $CsAgO_3$, $LaAgO_3$, $LiAgO_3$, $MgAgO_3$, $NaAgO_3$, $NdAgO_3$, $PrAgO_3$, $RbAgO_3$, $SmAgO_3$, $SrAgO_3$, $YAgO_3$
- **Ta - cluster:** $BaTaO_3$, $CaTaO_3$, $CeTaO_3$, $CsTaO_3$, $KTaO_3$, $LaTaO_3$, $LiTaO_3$, $MgTaO_3$, $NaTaO_3$, $NdTaO_3$, $PmTaO_3$, $SmTaO_3$, $SrTaO_3$, $YTaO_3$
- **W - cluster:** $BaWO_3$, $CaWO_3$, $CeWO_3$, $CsWO_3$, KWO_3, $LaWO_3$, $LiWO_3$, $MgWO_3$, $PmWO_3$, $PrWO_3$, $ScWO_3$, $SmWO_3$, $SrWO_3$, YWO_3
- **Pt - cluster:** $BaPtO_3$, $BePtO_3$, $CaPtO_3$, $CePtO_3$, $CsPtO_3$, $KPtO_3$, $LaPtO_3$, $LiPtO_3$, $MgPtO_3$, $NaPtO_3$, $NdPtO_3$, $PrPtO_3$, $RbPtO_3$, $SmPtO_3$, $YPtO_3$

- **Zn - cluster:** $BaZnO_3$, $BeZnO_3$, $KZnO_3$, $LaZnO_3$, $LiZnO_3$, $MgZnO_3$, $NaZnO_3$, $PmZnO_3$, $PrZnO_3$, $RbZnO_3$, $SrZnO_3$, $YZnO_3$
- **Cd - cluster:** $BaCdO_3$, $BeCdO_3$, $CaCdO_3$, $CeCdO_3$, $CsCdO_3$, $LiCdO_3$, $MgCdO_3$, $NdCdO_3$, $PmCdO_3$, $SmCdO_3$, $YCdO_3$
- **Al - cluster:** $BaAlO_3$, $BeAlO_3$, $CaAlO_3$, $LaAlO_3$, $LiAlO_3$, $NaAlO_3$, $NdAlO_3$, $PmAlO_3$, $PrAlO_3$, $ScAlO_3$, $SmAlO_3$, $SrAlO_3$, $YAlO_3$
- **Ga - cluster:** $BaGaO_3$, $CaGaO_3$, $CeGaO_3$, $KGaO_3$, $LaGaO_3$, $LiGaO_3$, $NaGaO_3$, $NdGaO_3$, $PmGaO_3$, $PrGaO_3$, $RbGaO_3$, $ScGaO_3$, $SmGaO_3$, $YGaO_3$
- **Sn - cluster:** $BeSnO_3$, $CeSnO_3$, $CsSnO_3$, $KSnO_3$, $LaSnO_3$, $MgSnO_3$, $NaSnO_3$, $NdSnO_3$, $PmSnO_3$, $PrSnO_3$, $RbSnO_3$, $ScSnO_3$, $SmSnO_3$, $YSnO_3$
- **Pb - cluster:** $BaPbO_3$, $BePbO_3$, $CaPbO_3$, $CsPbO_3$, $KPbO_3$, $LiPbO_3$, $MgPbO_3$, $NaPbO_3$, $NdPbO_3$, $PmPbO_3$, $PrPbO_3$, $RbPbO_3$, $SrPbO_3$, $YPbO_3$
- **Bi - cluster:** $BaBiO_3$, $CeBiO_3$, $KBiO_3$, $LiBiO_3$, $MgBiO_3$, $NaBiO_3$, $NdBiO_3$, $PmBiO_3$, $PrBiO_3$, $RbBiO_3$, $SmBiO_3$, $YBiO_3$
- **Ge - cluster:** $BaGeO_3$, $BeGeO_3$, $CeGeO_3$, $KGeO_3$, $LaGeO_3$, $NdGeO_3$, $PmGeO_3$, $PrGeO_3$, $RbGeO_3$, $SmGeO_3$, $SrGeO_3$, $YGeO_3$
- **Sb - cluster:** $BeSbO_3$, $CaSbO_3$, $CsSbO_3$, $KSbO_3$, $LaSbO_3$, $LiSbO_3$, $NaSbO_3$, $NdSbO_3$, $PmSbO_3$, $PrSbO_3$, $RbSbO_3$, $ScSbO_3$, $SmSbO_3$, $SrSbO_3$

A.3 GroupA Clustering:

Materials whose element *A* belongs to the same column in the periodic tabld are ascribed to the same cluster.

- **Group1 - cluster:** $CsAgO_3$, $CsCdO_3$, $CsCoO_3$, $CsCrO_3$, $CsFeO_3$, $CsMnO_3$, $CsMoO_3$, $CsNbO_3$, $CsNiO_3$, $CsPbO_3$, $CsPdO_3$, $CsPtO_3$, $CsRhO_3$, $CsRuO_3$, $CsSbO_3$, $CsSnO_3$, $CsTaO_3$, $CsVO_3$, $CsWO_3$, $CsZrO_3$, $KBiO_3$, $KCoO_3$, $KCrO_3$, $KCuO_3$, $KFeO_3$, $KGaO_3$, $KGeO_3$, $KMnO_3$, $KMoO_3$, $KNbO_3$, $KNiO_3$, $KPbO_3$, $KPdO_3$, $KPtO_3$, $KSbO_3$, $KSnO_3$, $KTaO_3$, $KTcO_3$, $KTiO_3$, KVO_3, KWO_3, $KZnO_3$, $KZrO_3$, $LiAgO_3$, $LiAlO_3$, $LiBiO_3$, $LiCdO_3$, $LiCoO_3$, $LiCrO_3$, $LiCuO_3$, $LiFeO_3$, $LiGaO_3$, $LiMoO_3$, $LiNbO_3$, $LiNiO_3$, $LiPbO_3$, $LiPdO_3$, $LiPtO_3$, $LiRhO_3$, $LiRuO_3$, $LiSbO_3$, $LiTaO_3$, $LiTcO_3$, $LiTiO_3$, $LiVO_3$, $LiWO_3$, $LiZnO_3$, $LiZrO_3$, $NaAgO_3$, $NaAlO_3$, $NaBiO_3$, $NaCoO_3$, $NaGaO_3$, $NaMnO_3$, $NaMoO_3$, $NaNbO_3$, $NaNiO_3$, $NaPbO_3$, $NaPdO_3$, $NaPtO_3$, $NaRhO_3$, $NaSbO_3$, $NaSnO_3$, $NaTaO_3$, $NaTiO_3$, $NaVO_3$, $NaZnO_3$, $NaZrO_3$, $RbAgO_3$, $RbBiO_3$, $RbCoO_3$, $RbCuO_3$, $RbFeO_3$, $RbGaO_3$, $RbGeO_3$, $RbMnO_3$, $RbMoO_3$, $RbNbO_3$, $RbNiO_3$, $RbPbO_3$, $RbPtO_3$, $RbRuO_3$, $RbSbO_3$, $RbSnO_3$, $RbTcO_3$, $RbTiO_3$, $RbVO_3$, $RbZnO_3$, $RbZrO_3$
- **Group2 - cluster:** $BaAgO_3$, $BaAlO_3$, $BaBiO_3$, $BaCdO_3$, $BaCoO_3$, $BaCrO_3$, $BaCuO_3$, $BaGaO_3$, $BaGeO_3$, $BaMnO_3$, $BaNiO_3$, $BaPbO_3$, $BaPdO_3$, $BaPtO_3$, $BaRhO_3$, $BaRuO_3$, $BaTaO_3$, $BaTcO_3$, $BaTiO_3$, $BaVO_3$, $BaWO_3$, $BaZnO_3$,

$BeAlO_3$, $BeCdO_3$, $BeCoO_3$, $BeCuO_3$, $BeFeO_3$, $BeGeO_3$, $BeMnO_3$, $BeNbO_3$, $BePbO_3$, $BePdO_3$, $BePtO_3$, $BeRuO_3$, $BeSbO_3$, $BeSnO_3$, $BeTiO_3$, $BeVO_3$, $BeZnO_3$, $BeZrO_3$, $CaAgO_3$, $CaAlO_3$, $CaCdO_3$, $CaCoO_3$, $CaCrO_3$, $CaCuO_3$, $CaFeO_3$, $CaGaO_3$, $CaMnO_3$, $CaMoO_3$, $CaNbO_3$, $CaPbO_3$, $CaPdO_3$, $CaPtO_3$, $CaRhO_3$, $CaSbO_3$, $CaTaO_3$, $CaTiO_3$, $CaWO_3$, $CaZrO_3$, $MgAgO_3$, $MgBiO_3$, $MgCdO_3$, $MgCuO_3$, $MgFeO_3$, $MgMnO_3$, $MgMoO_3$, $MgNbO_3$, $MgNiO_3$, $MgPbO_3$, $MgPdO_3$, $MgPtO_3$, $MgSnO_3$, $MgTaO_3$, $MgTcO_3$, $MgTiO_3$, $MgVO_3$, $MgWO_3$, $MgZnO_3$, $MgZrO_3$, $SrAgO_3$, $SrAlO_3$, $SrCoO_3$, $SrCrO_3$, $SrFeO_3$, $SrGeO_3$, $SrMnO_3$, $SrMoO_3$, $SrNbO_3$, $SrNiO_3$, $SrPbO_3$, $SrRhO_3$, $SrSbO_3$, $SrTaO_3$, $SrTcO_3$, $SrVO_3$, $SrWO_3$, $SrZnO_3$, $SrZrO_3$

- **Group3 - cluster:** $LaAgO_3$, $LaAlO_3$, $LaCoO_3$, $LaCrO_3$, $LaCuO_3$, $LaFeO_3$, $LaGaO_3$, $LaGeO_3$, $LaMnO_3$, $LaMoO_3$, $LaNbO_3$, $LaNiO_3$, $LaPdO_3$, $LaPtO_3$, $LaRuO_3$, $LaSbO_3$, $LaSnO_3$, $LaTaO_3$, $LaTiO_3$, $LaVO_3$, $LaWO_3$, $LaZnO_3$, $LaZrO_3$, $ScAlO_3$, $ScCoO_3$, $ScCrO_3$, $ScCuO_3$, $ScGaO_3$, $ScMnO_3$, $ScMoO_3$, $ScNbO_3$, $ScNiO_3$, $ScPdO_3$, $ScRhO_3$, $ScRuO_3$, $ScSbO_3$, $ScSnO_3$, $ScTcO_3$, $ScTiO_3$, $ScWO_3$, $ScZrO_3$, $YAgO_3$, $YAlO_3$, $YBiO_3$, $YCdO_3$, $YCoO_3$, $YCrO_3$, $YCuO_3$, $YFeO_3$, $YGaO_3$, $YGeO_3$, $YMnO_3$, $YMoO_3$, $YNiO_3$, $YPbO_3$, $YPtO_3$, $YRhO_3$, $YRuO_3$, $YSnO_3$, $YTaO_3$, $YTcO_3$, $YTiO_3$, YVO_3, YWO_3, $YZnO_3$, $YZrO_3$
- **Ce - cluster:** $CeAgO_3$, $CeBiO_3$, $CeCdO_3$, $CeCoO_3$, $CeCrO_3$, $CeGaO_3$, $CeGeO_3$, $CeMnO_3$, $CeMoO_3$, $CeNbO_3$, $CeNiO_3$, $CePtO_3$, $CeRhO_3$, $CeRuO_3$, $CeSnO_3$, $CeTaO_3$, $CeTcO_3$, $CeTiO_3$, $CeVO_3$, $CeWO_3$, $CeZrO_3$
- **Pr - cluster:** $PrAgO_3$, $PrAlO_3$, $PrBiO_3$, $PrCoO_3$, $PrCrO_3$, $PrFeO_3$, $PrGaO_3$, $PrGeO_3$, $PrMnO_3$, $PrNbO_3$, $PrNiO_3$, $PrPbO_3$, $PrPtO_3$, $PrRuO_3$, $PrSbO_3$, $PrSnO_3$, $PrTcO_3$, $PrTiO_3$, $PrVO_3$, $PrWO_3$, $PrZnO_3$
- **Nd - cluster:** $NdAgO_3$, $NdAlO_3$, $NdBiO_3$, $NdCdO_3$, $NdCrO_3$, $NdGaO_3$, $NdGeO_3$, $NdMnO_3$, $NdMoO_3$, $NdNbO_3$, $NdNiO_3$, $NdPbO_3$, $NdPtO_3$, $NdRuO_3$, $NdSbO_3$, $NdSnO_3$, $NdTaO_3$, $NdTiO_3$, $NdVO_3$, $NdZrO_3$
- **Pm - cluster:** $PmAlO_3$, $PmBiO_3$, $PmCdO_3$, $PmCrO_3$, $PmCuO_3$, $PmFeO_3$, $PmGaO_3$, $PmGeO_3$, $PmMnO_3$, $PmMoO_3$, $PmNbO_3$, $PmNiO_3$, $PmPbO_3$, $PmPdO_3$, $PmSbO_3$, $PmSnO_3$, $PmTaO_3$, $PmTcO_3$, $PmWO_3$, $PmZnO_3$, $PmZrO_3$
- **Sm - cluster:** $SmAgO_3$, $SmAlO_3$, $SmBiO_3$, $SmCdO_3$, $SmCrO_3$, $SmFeO_3$, $SmGaO_3$, $SmGeO_3$, $SmNbO_3$, $SmNiO_3$, $SmPtO_3$, $SmRhO_3$, $SmRuO_3$, $SmSbO_3$, $SmSnO_3$, $SmTaO_3$, $SmTcO_3$, $SmTiO_3$, $SmVO_3$, $SmWO_3$, $SmZrO_3$

A.4 GroupB Clustering

Materials whose element B belongs to the same column in the periodic tabld are ascribed to the same cluster.

- **Group4 - cluster:** $BaTiO_3$, $BeTiO_3$, $BeZrO_3$, $CaTiO_3$, $CaZrO_3$, $CeTiO_3$, $CeZrO_3$, $CsZrO_3$, $KTiO_3$, $KZrO_3$, $LaTiO_3$, $LaZrO_3$, $LiTiO_3$, $LiZrO_3$, $MgTiO_3$, $MgZrO_3$, $NaTiO_3$, $NaZrO_3$, $NdTiO_3$, $NdZrO_3$, $PmZrO_3$, $PrTiO_3$, $RbTiO_3$, $RbZrO_3$, $ScTiO_3$, $ScZrO_3$, $SmTiO_3$, $SmZrO_3$, $SrZrO_3$, $YTiO_3$, $YZrO_3$

- **Group5 - cluster:** $BaTaO_3$, $BaVO_3$, $BeNbO_3$, $BeVO_3$, $CaNbO_3$, $CaTaO_3$, $CeNbO_3$, $CeTaO_3$, $CeVO_3$, $CsNbO_3$, $CsTaO_3$, $CsVO_3$, $KNbO_3$, $KTaO_3$, KVO_3, $LaNbO_3$, $LaTaO_3$, $LaVO_3$, $LiNbO_3$, $LiTaO_3$, $LiVO_3$, $MgNbO_3$, $MgTaO_3$, $MgVO_3$, $NaNbO_3$, $NaTaO_3$, $NaVO_3$, $NdNbO_3$, $NdTaO_3$, $NdVO_3$, $PmNbO_3$, $PmTaO_3$, $PrNbO_3$, $PrVO_3$, $RbNbO_3$, $RbVO_3$, $ScNbO_3$, $SmNbO_3$, $SmTaO_3$, $SmVO_3$, $SrNbO_3$, $SrTaO_3$, $SrVO_3$, $YTaO_3$, YVO_3
- **Group6 - cluster:** $BaCrO_3$, $BaWO_3$, $CaCrO_3$, $CaMoO_3$, $CaWO_3$, $CeCrO_3$, $CeMoO_3$, $CeWO_3$, $CsCrO_3$, $CsMoO_3$, $CsWO_3$, $KCrO_3$, $KMoO_3$, KWO_3, $LaCrO_3$, $LaMoO_3$, $LaWO_3$, $LiCrO_3$, $LiMoO_3$, $LiWO_3$, $MgMoO_3$, $MgWO_3$, $NaMoO_3$, $NdCrO_3$, $NdMoO_3$, $PmCrO_3$, $PmMoO_3$, $PmWO_3$, $PrCrO_3$, $PrWO_3$, $RbMoO_3$, $ScCrO_3$, $ScMoO_3$, $ScWO_3$, $SmCrO_3$, $SmWO_3$, $SrCrO_3$, $SrMoO_3$, $SrWO_3$, $YCrO_3$, $YMoO_3$, YWO_3
- **Group7 - cluster:** $BaMnO_3$, $BaTcO_3$, $BeMnO_3$, $CaMnO_3$, $CeMnO_3$, $CeTcO_3$, $CsMnO_3$, $KMnO_3$, $KTcO_3$, $LaMnO_3$, $LiTcO_3$, $MgMnO_3$, $MgTcO_3$, $NaMnO_3$, $NdMnO_3$, $PmMnO_3$, $PmTcO_3$, $PrMnO_3$, $PrTcO_3$, $RbMnO_3$, $RbTcO_3$, $ScMnO_3$, $ScTcO_3$, $SmTcO_3$, $SrMnO_3$, $SrTcO_3$, $YMnO_3$, $YTcO_3$
- **Group8 - cluster:** $BaRuO_3$, $BeFeO_3$, $BeRuO_3$, $CaFeO_3$, $CeRuO_3$, $CsFeO_3$, $CsRuO_3$, $KFeO_3$, $LaFeO_3$, $LaRuO_3$, $LiFeO_3$, $LiRuO_3$, $MgFeO_3$, $NdRuO_3$, $PmFeO_3$, $PrFeO_3$, $PrRuO_3$, $RbFeO_3$, $RbRuO_3$, $ScRuO_3$, $SmFeO_3$, $SmRuO_3$, $SrFeO_3$, $YFeO_3$, $YRuO_3$
- **Group9 - cluster:** $BaCoO_3$, $BaRhO_3$, $BeCoO_3$, $CaCoO_3$, $CaRhO_3$, $CeCoO_3$, $CeRhO_3$, $CsCoO_3$, $CsRhO_3$, $KCoO_3$, $LaCoO_3$, $LiCoO_3$, $LiRhO_3$, $NaCoO_3$, $NaRhO_3$, $PrCoO_3$, $RbCoO_3$, $ScCoO_3$, $ScRhO_3$, $SmRhO_3$, $SrCoO_3$, $SrRhO_3$, $YCoO_3$, $YRhO_3$
- **Group10 - cluster:** $BaNiO_3$, $BaPdO_3$, $BaPtO_3$, $BePdO_3$, $BePtO_3$, $CaPdO_3$, $CaPtO_3$, $CeNiO_3$, $CePtO_3$, $CsNiO_3$, $CsPdO_3$, $CsPtO_3$, $KNiO_3$, $KPdO_3$, $KPtO_3$, $LaNiO_3$, $LaPdO_3$, $LaPtO_3$, $LiNiO_3$, $LiPdO_3$, $LiPtO_3$, $MgNiO_3$, $MgPdO_3$, $MgPtO_3$, $NaNiO_3$, $NaPdO_3$, $NaPtO_3$, $NdNiO_3$, $NdPtO_3$, $PmNiO_3$, $PmPdO_3$, $PrNiO_3$, $PrPtO_3$, $RbNiO_3$, $RbPtO_3$, $ScNiO_3$, $ScPdO_3$, $SmNiO_3$, $SmPtO_3$, $SrNiO_3$, $YNiO_3$, $YPtO_3$
- **Group11 - cluster:** $BaAgO_3$, $BaCuO_3$, $BeCuO_3$, $CaAgO_3$, $CaCuO_3$, $CeAgO_3$, $CsAgO_3$, $KCuO_3$, $LaAgO_3$, $LaCuO_3$, $LiAgO_3$, $LiCuO_3$, $MgAgO_3$, $MgCuO_3$, $NaAgO_3$, $NdAgO_3$, $PmCuO_3$, $PrAgO_3$, $RbAgO_3$, $RbCuO_3$, $ScCuO_3$, $SmAgO_3$, $SrAgO_3$, $YAgO_3$, $YCuO_3$
- **Group12 - cluster:** $BaCdO_3$, $BaZnO_3$, $BeCdO_3$, $BeZnO_3$, $CaCdO_3$, $CeCdO_3$, $CsCdO_3$, $KZnO_3$, $LaZnO_3$, $LiCdO_3$, $LiZnO_3$, $MgCdO_3$, $MgZnO_3$, $NaZnO_3$, $NdCdO_3$, $PmCdO_3$, $PmZnO_3$, $PrZnO_3$, $RbZnO_3$, $SmCdO_3$, $SrZnO_3$, $YCdO_3$, $YZnO_3$
- **Group13 - cluster:** $BaAlO_3$, $BaGaO_3$, $BeAlO_3$, $CaAlO_3$, $CaGaO_3$, $CeGaO_3$, $KGaO_3$, $LaAlO_3$, $LaGaO_3$, $LiAlO_3$, $LiGaO_3$, $NaAlO_3$, $NaGaO_3$, $NdAlO_3$, $NdGaO_3$, $PmAlO_3$, $PmGaO_3$, $PrAlO_3$, $PrGaO_3$, $RbGaO_3$, $ScAlO_3$, $ScGaO_3$, $SmAlO_3$, $SmGaO_3$, $SrAlO_3$, $YAlO_3$, $YGaO_3$

- **Group14 - cluster:** $BaGeO_3$, $BaPbO_3$, $BeGeO_3$, $BePbO_3$, $BeSnO_3$, $CaPbO_3$, $CeGeO_3$, $CeSnO_3$, $CsPbO_3$, $CsSnO_3$, $KGeO_3$, $KPbO_3$, $KSnO_3$, $LaGeO_3$, $LaSnO_3$, $LiPbO_3$, $MgPbO_3$, $MgSnO_3$, $NaPbO_3$, $NaSnO_3$, $NdGeO_3$, $NdPbO_3$, $NdSnO_3$, $PmGeO_3$, $PmPbO_3$, $PmSnO_3$, $PrGeO_3$, $PrPbO_3$, $PrSnO_3$, $RbGeO_3$, $RbPbO_3$, $RbSnO_3$, $ScSnO_3$, $SmGeO_3$, $SmSnO_3$, $SrGeO_3$, $SrPbO_3$, $YGeO_3$, $YPbO_3$, $YSnO_3$
- **Group15 - cluster:** $BaBiO_3$, $BeSbO_3$, $CaSbO_3$, $CeBiO_3$, $CsSbO_3$, $KBiO_3$, $KSbO_3$, $LaSbO_3$, $LiBiO_3$, $LiSbO_3$, $MgBiO_3$, $NaBiO_3$, $NaSbO_3$, $NdBiO_3$, $NdSbO_3$, $PmBiO_3$, $PmSbO_3$, $PrBiO_3$, $PrSbO_3$, $RbBiO_3$, $RbSbO_3$, $ScSbO_3$, $SmBiO_3$, $SmSbO_3$, $SrSbO_3$, $YBiO_3$

A.5 TypeA Clustering

The type of element A of a material determines its cluster membership.

- **Alkali metals - cluster:** $CsAgO_3$, $CsCdO_3$, $CsCoO_3$, $CsCrO_3$, $CsFeO_3$, $CsMnO_3$, $CsMoO_3$, $CsNbO_3$, $CsNiO_3$, $CsPbO_3$, $CsPdO_3$, $CsPtO_3$, $CsRhO_3$, $CsRuO_3$, $CsSbO_3$, $CsSnO_3$, $CsTaO_3$, $CsVO_3$, $CsWO_3$, $CsZrO_3$, $KBiO_3$, $KCoO_3$, $KCrO_3$, $KCuO_3$, $KFeO_3$, $KGaO_3$, $KGeO_3$, $KMnO_3$, $KMoO_3$, $KNbO_3$, $KNiO_3$, $KPbO_3$, $KPdO_3$, $KPtO_3$, $KSbO_3$, $KSnO_3$, $KTaO_3$, $KTcO_3$, $KTiO_3$, KVO_3, KWO_3, $KZnO_3$, $KZrO_3$, $LiAgO_3$, $LiAlO_3$, $LiBiO_3$, $LiCdO_3$, $LiCoO_3$, $LiCrO_3$, $LiCuO_3$, $LiFeO_3$, $LiGaO_3$, $LiMoO_3$, $LiNbO_3$, $LiNiO_3$, $LiPbO_3$, $LiPdO_3$, $LiPtO_3$, $LiRhO_3$, $LiRuO_3$, $LiSbO_3$, $LiTaO_3$, $LiTcO_3$, $LiTiO_3$, $LiVO_3$, $LiWO_3$, $LiZnO_3$, $LiZrO_3$, $NaAgO_3$, $NaAlO_3$, $NaBiO_3$, $NaCoO_3$, $NaGaO_3$, $NaMnO_3$, $NaMoO_3$, $NaNbO_3$, $NaNiO_3$, $NaPbO_3$, $NaPdO_3$, $NaPtO_3$, $NaRhO_3$, $NaSbO_3$, $NaSnO_3$, $NaTaO_3$, $NaTiO_3$, $NaVO_3$, $NaZnO_3$, $NaZrO_3$, $RbAgO_3$, $RbBiO_3$, $RbCoO_3$, $RbCuO_3$, $RbFeO_3$, $RbGaO_3$, $RbGeO_3$, $RbMnO_3$, $RbMoO_3$, $RbNbO_3$, $RbNiO_3$, $RbPbO_3$, $RbPtO_3$, $RbRuO_3$, $RbSbO_3$, $RbSnO_3$, $RbTcO_3$, $RbTiO_3$, $RbVO_3$, $RbZnO_3$, $RbZrO_3$
- **Alkaline earth metal - cluster:** $BaAgO_3$, $BaAlO_3$, $BaBiO_3$, $BaCdO_3$, $BaCoO_3$, $BaCrO_3$, $BaCuO_3$, $BaGaO_3$, $BaGeO_3$, $BaMnO_3$, $BaNiO_3$, $BaPbO_3$, $BaPdO_3$, $BaPtO_3$, $BaRhO_3$, $BaRuO_3$, $BaTaO_3$, $BaTcO_3$, $BaTiO_3$, $BaVO_3$, $BaWO_3$, $BaZnO_3$, $BeAlO_3$, $BeCdO_3$, $BeCoO_3$, $BeCuO_3$, $BeFeO_3$, $BeGeO_3$, $BeMnO_3$, $BeNbO_3$, $BePbO_3$, $BePdO_3$, $BePtO_3$, $BeRuO_3$, $BeSbO_3$, $BeSnO_3$, $BeTiO_3$, $BeVO_3$, $BeZnO_3$, $BeZrO_3$, $CaAgO_3$, $CaAlO_3$, $CaCdO_3$, $CaCoO_3$, $CaCrO_3$, $CaCuO_3$, $CaFeO_3$, $CaGaO_3$, $CaMnO_3$, $CaMoO_3$, $CaNbO_3$, $CaPbO_3$, $CaPdO_3$, $CaPtO_3$, $CaRhO_3$, $CaSbO_3$, $CaTaO_3$, $CaTiO_3$, $CaWO_3$, $CaZrO_3$, $MgAgO_3$, $MgBiO_3$, $MgCdO_3$, $MgCuO_3$, $MgFeO_3$, $MgMnO_3$, $MgMoO_3$, $MgNbO_3$, $MgNiO_3$, $MgPbO_3$, $MgPdO_3$, $MgPtO_3$, $MgSnO_3$, $MgTaO_3$, $MgTcO_3$, $MgTiO_3$, $MgVO_3$, $MgWO_3$, $MgZnO_3$, $MgZrO_3$, $SrAgO_3$, $SrAlO_3$, $SrCoO_3$, $SrCrO_3$, $SrFeO_3$, $SrGeO_3$, $SrMnO_3$, $SrMoO_3$, $SrNbO_3$, $SrNiO_3$, $SrPbO_3$, $SrRhO_3$, $SrSbO_3$, $SrTaO_3$, $SrTcO_3$, $SrVO_3$, $SrWO_3$, $SrZnO_3$, $SrZrO_3$
- **Lanthanide - cluster:** $CeAgO_3$, $CeBiO_3$, $CeCdO_3$, $CeCoO_3$, $CeCrO_3$, $CeGaO_3$, $CeGeO_3$, $CeMnO_3$, $CeMoO_3$, $CeNbO_3$, $CeNiO_3$, $CePtO_3$, $CeRhO_3$, $CeRuO_3$, $CeSnO_3$, $CeTaO_3$, $CeTcO_3$, $CeTiO_3$, $CeVO_3$, $CeWO_3$, $CeZrO_3$, $LaAgO_3$, $LaAlO_3$,

$LaCoO_3$, $LaCrO_3$, $LaCuO_3$, $LaFeO_3$, $LaGaO_3$, $LaGeO_3$, $LaMnO_3$, $LaMoO_3$, $LaNbO_3$, $LaNiO_3$, $LaPdO_3$, $LaPtO_3$, $LaRuO_3$, $LaSbO_3$, $LaSnO_3$, $LaTaO_3$, $LaTiO_3$, $LaVO_3$, $LaWO_3$, $LaZnO_3$, $LaZrO_3$, $NdAgO_3$, $NdAlO_3$, $NdBiO_3$, $NdCdO_3$, $NdCrO_3$, $NdGaO_3$, $NdGeO_3$, $NdMnO_3$, $NdMoO_3$, $NdNbO_3$, $NdNiO_3$, $NdPbO_3$, $NdPtO_3$, $NdRuO_3$, $NdSbO_3$, $NdSnO_3$, $NdTaO_3$, $NdTiO_3$, $NdVO_3$, $NdZrO_3$, $PmAlO_3$, $PmBiO_3$, $PmCdO_3$, $PmCrO_3$, $PmCuO_3$, $PmFeO_3$, $PmGaO_3$, $PmGeO_3$, $PmMnO_3$, $PmMoO_3$, $PmNbO_3$, $PmNiO_3$, $PmPbO_3$, $PmPdO_3$, $PmSbO_3$, $PmSnO_3$, $PmTaO_3$, $PmTcO_3$, $PmWO_3$, $PmZnO_3$, $PmZrO_3$, $PrAgO_3$, $PrAlO_3$, $PrBiO_3$, $PrCoO_3$, $PrCrO_3$, $PrFeO_3$, $PrGaO_3$, $PrGeO_3$, $PrMnO_3$, $PrNbO_3$, $PrNiO_3$, $PrPbO_3$, $PrPtO_3$, $PrRuO_3$, $PrSbO_3$, $PrSnO_3$, $PrTcO_3$, $PrTiO_3$, $PrVO_3$, $PrWO_3$, $PrZnO_3$, $SmAgO_3$, $SmAlO_3$, $SmBiO_3$, $SmCdO_3$, $SmCrO_3$, $SmFeO_3$, $SmGaO_3$, $SmGeO_3$, $SmNbO_3$, $SmNiO_3$, $SmPtO_3$, $SmRhO_3$, $SmRuO_3$, $SmSbO_3$, $SmSnO_3$, $SmTaO_3$, $SmTcO_3$, $SmTiO_3$, $SmVO_3$, $SmWO_3$, $SmZrO_3$

- **Transition metal - cluster:** $ScAlO_3$, $ScCoO_3$, $ScCrO_3$, $ScCuO_3$, $ScGaO_3$, $ScMnO_3$, $ScMoO_3$, $ScNbO_3$, $ScNiO_3$, $ScPdO_3$, $ScRhO_3$, $ScRuO_3$, $ScSbO_3$, $ScSnO_3$, $ScTcO_3$, $ScTiO_3$, $ScWO_3$, $ScZrO_3$, $YAgO_3$, $YAlO_3$, $YBiO_3$, $YCdO_3$, $YCoO_3$, $YCrO_3$, $YCuO_3$, $YFeO_3$, $YGaO_3$, $YGeO_3$, $YMnO_3$, $YMoO_3$, $YNiO_3$, $YPbO_3$, $YPtO_3$, $YRhO_3$, $YRuO_3$, $YSnO_3$, $YTaO_3$, $YTcO_3$, $YTiO_3$, YVO_3, YWO_3, $YZnO_3$, $YZrO_3$

A.6 TypeB Clustering

The type of element B of a material determines its cluster membership.

- **Transition metal - cluster:** $BaAgO_3$, $BaCoO_3$, $BaCrO_3$, $BaCuO_3$, $BaMnO_3$, $BaNiO_3$, $BaPdO_3$, $BaPtO_3$, $BaRhO_3$, $BaRuO_3$, $BaTaO_3$, $BaTcO_3$, $BaTiO_3$, $BaVO_3$, $BaWO_3$, $BeCoO_3$, $BeCuO_3$, $BeFeO_3$, $BeMnO_3$, $BeNbO_3$, $BePdO_3$, $BePtO_3$, $BeRuO_3$, $BeTiO_3$, $BeVO_3$, $BeZrO_3$, $CaAgO_3$, $CaCoO_3$, $CaCrO_3$, $CaCuO_3$, $CaFeO_3$, $CaMnO_3$, $CaMoO_3$, $CaNbO_3$, $CaPdO_3$, $CaPtO_3$, $CaRhO_3$, $CaTaO_3$, $CaTiO_3$, $CaWO_3$, $CaZrO_3$, $CeAgO_3$, $CeCoO_3$, $CeCrO_3$, $CeMnO_3$, $CeMoO_3$, $CeNbO_3$, $CeNiO_3$, $CePtO_3$, $CeRhO_3$, $CeRuO_3$, $CeTaO_3$, $CeTcO_3$, $CeTiO_3$, $CeVO_3$, $CeWO_3$, $CeZrO_3$, $CsAgO_3$, $CsCoO_3$, $CsCrO_3$, $CsFeO_3$, $CsMnO_3$, $CsMoO_3$, $CsNbO_3$, $CsNiO_3$, $CsPdO_3$, $CsPtO_3$, $CsRhO_3$, $CsRuO_3$, $CsTaO_3$, $CsVO_3$, $CsWO_3$, $CsZrO_3$, $KCoO_3$, $KCrO_3$, $KCuO_3$, $KFeO_3$, $KMnO_3$, $KMoO_3$, $KNbO_3$, $KNiO_3$, $KPdO_3$, $KPtO_3$, $KTaO_3$, $KTcO_3$, $KTiO_3$, KVO_3, KWO_3, $KZrO_3$, $LaAgO_3$, $LaCoO_3$, $LaCrO_3$, $LaCuO_3$, $LaFeO_3$, $LaMnO_3$, $LaMoO_3$, $LaNbO_3$, $LaNiO_3$, $LaPdO_3$, $LaPtO_3$, $LaRuO_3$, $LaTaO_3$, $LaTiO_3$, $LaVO_3$, $LaWO_3$, $LaZrO_3$, $LiAgO_3$, $LiCoO_3$, $LiCrO_3$, $LiCuO_3$, $LiFeO_3$, $LiMoO_3$, $LiNbO_3$, $LiNiO_3$, $LiPdO_3$, $LiPtO_3$, $LiRhO_3$, $LiRuO_3$, $LiTaO_3$, $LiTcO_3$, $LiTiO_3$, $LiVO_3$, $LiWO_3$, $LiZrO_3$, $MgAgO_3$, $MgCuO_3$, $MgFeO_3$, $MgMnO_3$, $MgMoO_3$, $MgNbO_3$, $MgNiO_3$, $MgPdO_3$, $MgPtO_3$, $MgTaO_3$, $MgTcO_3$, $MgTiO_3$, $MgVO_3$, $MgWO_3$, $MgZrO_3$, $NaAgO_3$, $NaCoO_3$, $NaMnO_3$, $NaMoO_3$, $NaNbO_3$, $NaNiO_3$, $NaPdO_3$, $NaPtO_3$, $NaRhO_3$, $NaTaO_3$, $NaTiO_3$, $NaVO_3$, $NaZrO_3$, $NdAgO_3$, $NdCrO_3$, $NdMnO_3$, $NdMoO_3$, $NdNbO_3$, $NdNiO_3$, $NdPtO_3$, $NdRuO_3$, $NdTaO_3$, $NdTiO_3$, $NdVO_3$, $NdZrO_3$, $PmCrO_3$, $PmCuO_3$, $PmFeO_3$, $PmMnO_3$, $PmMoO_3$, $PmNbO_3$, $PmNiO_3$, $PmPdO_3$, $PmTaO_3$, $PmTcO_3$, $PmWO_3$, $PmZrO_3$, $PrAgO_3$, $PrCoO_3$, $PrCrO_3$,

$PrFeO_3$, $PrMnO_3$, $PrNbO_3$, $PrNiO_3$, $PrPtO_3$, $PrRuO_3$, $PrTcO_3$, $PrTiO_3$, $PrVO_3$, $PrWO_3$, $RbAgO_3$, $RbCoO_3$, $RbCuO_3$, $RbFeO_3$, $RbMnO_3$, $RbMoO_3$, $RbNbO_3$, $RbNiO_3$, $RbPtO_3$, $RbRuO_3$, $RbTcO_3$, $RbTiO_3$, $RbVO_3$, $RbZrO_3$, $ScCoO_3$, $ScCrO_3$, $ScCuO_3$, $ScMnO_3$, $ScMoO_3$, $ScNbO_3$, $ScNiO_3$, $ScPdO_3$, $ScRhO_3$, $ScRuO_3$, $ScTcO_3$, $ScTiO_3$, $ScWO_3$, $ScZrO_3$, $SmAgO_3$, $SmCrO_3$, $SmFeO_3$, $SmNbO_3$, $SmNiO_3$, $SmPtO_3$, $SmRhO_3$, $SmRuO_3$, $SmTaO_3$, $SmTcO_3$, $SmTiO_3$, $SmVO_3$, $SmWO_3$, $SmZrO_3$, $SrAgO_3$, $SrCoO_3$, $SrCrO_3$, $SrFeO_3$, $SrMnO_3$, $SrMoO_3$, $SrNbO_3$, $SrNiO_3$, $SrRhO_3$, $SrTaO_3$, $SrTcO_3$, $SrVO_3$, $SrWO_3$, $SrZrO_3$, $YAgO_3$, $YCoO_3$, $YCrO_3$, $YCuO_3$, $YFeO_3$, $YMnO_3$, $YMoO_3$, $YNiO_3$, $YPtO_3$, $YRhO_3$, $YRuO_3$, $YTaO_3$, $YTcO_3$, $YTiO_3$, YVO_3, YWO_3, $YZrO_3$

- **Other metal - cluster:** $BaAlO_3$, $BaBiO_3$, $BaCdO_3$, $BaGaO_3$, $BaPbO_3$, $BaZnO_3$, $BeAlO_3$, $BeCdO_3$, $BePbO_3$, $BeSnO_3$, $BeZnO_3$, $CaAlO_3$, $CaCdO_3$, $CaGaO_3$, $CaPbO_3$, $CeBiO_3$, $CeCdO_3$, $CeGaO_3$, $CeSnO_3$, $CsCdO_3$, $CsPbO_3$, $CsSnO_3$, $KBiO_3$, $KGaO_3$, $KPbO_3$, $KSnO_3$, $KZnO_3$, $LaAlO_3$, $LaGaO_3$, $LaSnO_3$, $LaZnO_3$, $LiAlO_3$, $LiBiO_3$, $LiCdO_3$, $LiGaO_3$, $LiPbO_3$, $LiZnO_3$, $MgBiO_3$, $MgCdO_3$, $MgPbO_3$, $MgSnO_3$, $MgZnO_3$, $NaAlO_3$, $NaBiO_3$, $NaGaO_3$, $NaPbO_3$, $NaSnO_3$, $NaZnO_3$, $NdAlO_3$, $NdBiO_3$, $NdCdO_3$, $NdGaO_3$, $NdPbO_3$, $NdSnO_3$, $PmAlO_3$, $PmBiO_3$, $PmCdO_3$, $PmGaO_3$, $PmPbO_3$, $PmSnO_3$, $PmZnO_3$, $PrAlO_3$, $PrBiO_3$, $PrGaO_3$, $PrPbO_3$, $PrSnO_3$, $PrZnO_3$, $RbBiO_3$, $RbGaO_3$, $RbPbO_3$, $RbSnO_3$, $RbZnO_3$, $ScAlO_3$, $ScGaO_3$, $ScSnO_3$, $SmAlO_3$, $SmBiO_3$, $SmCdO_3$, $SmGaO_3$, $SmSnO_3$, $SrAlO_3$, $SrPbO_3$, $SrZnO_3$, $YAlO_3$, $YBiO_3$, $YCdO_3$, $YGaO_3$, $YPbO_3$, $YSnO_3$, $YZnO_3$

- **Metalloid - cluster:** $BaGeO_3$, $BeGeO_3$, $BeSbO_3$, $CaSbO_3$, $CeGeO_3$, $CsSbO_3$, $KGeO_3$, $KSbO_3$, $LaGeO_3$, $LaSbO_3$, $LiSbO_3$, $NaSbO_3$, $NdGeO_3$, $NdSbO_3$, $PmGeO_3$, $PmSbO_3$, $PrGeO_3$, $PrSbO_3$, $RbGeO_3$, $RbSbO_3$, $ScSbO_3$, $SmGeO_3$, $SmSbO_3$, $SrGeO_3$, $SrSbO_3$, $YGeO_3$

A.7 TypeAB Clustering

The type of element A as well as the one of element B of a material determine its cluster membership.

- **Alkali metal / transition metal - cluster:** $CsAgO_3$, $CsCoO_3$, $CsCrO_3$, $CsFeO_3$, $CsMnO_3$, $CsMoO_3$, $CsNbO_3$, $CsNiO_3$, $CsPdO_3$, $CsPtO_3$, $CsRhO_3$, $CsRuO_3$, $CsTaO_3$, $CsVO_3$, $CsWO_3$, $CsZrO_3$, $KCoO_3$, $KCrO_3$, $KCuO_3$, $KFeO_3$, $KMnO_3$, $KMoO_3$, $KNbO_3$, $KNiO_3$, $KPdO_3$, $KPtO_3$, $KTaO_3$, $KTcO_3$, $KTiO_3$, KVO_3, KWO_3, $KZrO_3$, $LiAgO_3$, $LiCoO_3$, $LiCrO_3$, $LiCuO_3$, $LiFeO_3$, $LiMoO_3$, $LiNbO_3$, $LiNiO_3$, $LiPdO_3$, $LiPtO_3$, $LiRhO_3$, $LiRuO_3$, $LiTaO_3$, $LiTcO_3$, $LiTiO_3$, $LiVO_3$, $LiWO_3$, $LiZrO_3$, $NaAgO_3$, $NaCoO_3$, $NaMnO_3$, $NaMoO_3$, $NaNbO_3$, $NaNiO_3$, $NaPdO_3$, $NaPtO_3$, $NaRhO_3$, $NaTaO_3$, $NaTiO_3$, $NaVO_3$, $NaZrO_3$, $RbAgO_3$, $RbCoO_3$, $RbCuO_3$, $RbFeO_3$, $RbMnO_3$, $RbMoO_3$, $RbNbO_3$, $RbNiO_3$, $RbPtO_3$, $RbRuO_3$, $RbTcO_3$, $RbTiO_3$, $RbVO_3$, $RbZrO_3$

- **Alkali metal / other metal - cluster:** $CsCdO_3$, $CsPbO_3$, $CsSnO_3$, $KBiO_3$, $KGaO_3$, $KPbO_3$, $KSnO_3$, $KZnO_3$, $LiAlO_3$, $LiBiO_3$, $LiCdO_3$, $LiGaO_3$, $LiPbO_3$,

$LiZnO_3$, $NaAlO_3$, $NaBiO_3$, $NaGaO_3$, $NaPbO_3$, $NaSnO_3$, $NaZnO_3$, $RbBiO_3$, $RbGaO_3$, $RbPbO_3$, $RbSnO_3$, $RbZnO_3$

- **Alkali metal / metalloid - cluster:** $CsSbO_3$, $KGeO_3$, $KSbO_3$, $LiSbO_3$, $NaSbO_3$, $RbGeO_3$, $RbSbO_3$
- **Alkaline earth metal / transition metal - cluster:** $BaAgO_3$, $BaCoO_3$, $BaCrO_3$, $BaCuO_3$, $BaMnO_3$, $BaNiO_3$, $BaPdO_3$, $BaPtO_3$, $BaRhO_3$, $BaRuO_3$, $BaTaO_3$, $BaTcO_3$, $BaTiO_3$, $BaVO_3$, $BaWO_3$, $BeCoO_3$, $BeCuO_3$, $BeFeO_3$, $BeMnO_3$, $BeNbO_3$, $BePdO_3$, $BePtO_3$, $BeRuO_3$, $BeTiO_3$, $BeVO_3$, $BeZrO_3$, $CaAgO_3$, $CaCoO_3$, $CaCrO_3$, $CaCuO_3$, $CaFeO_3$, $CaMnO_3$, $CaMoO_3$, $CaNbO_3$, $CaPdO_3$, $CaPtO_3$, $CaRhO_3$, $CaTaO_3$, $CaTiO_3$, $CaWO_3$, $CaZrO_3$, $MgAgO_3$, $MgCuO_3$, $MgFeO_3$, $MgMnO_3$, $MgMoO_3$, $MgNbO_3$, $MgNiO_3$, $MgPdO_3$, $MgPtO_3$, $MgTaO_3$, $MgTcO_3$, $MgTiO_3$, $MgVO_3$, $MgWO_3$, $MgZrO_3$, $SrAgO_3$, $SrCoO_3$, $SrCrO_3$, $SrFeO_3$, $SrMnO_3$, $SrMoO_3$, $SrNbO_3$, $SrNiO_3$, $SrRhO_3$, $SrTaO_3$, $SrTcO_3$, $SrVO_3$, $SrWO_3$, $SrZrO_3$
- **Alkaline earth metal / other metal - cluster:** $BaAlO_3$, $BaBiO_3$, $BaCdO_3$, $BaGaO_3$, $BaPbO_3$, $BaZnO_3$, $BeAlO_3$, $BeCdO_3$, $BePbO_3$, $BeSnO_3$, $BeZnO_3$, $CaAlO_3$, $CaCdO_3$, $CaGaO_3$, $CaPbO_3$, $MgBiO_3$, $MgCdO_3$, $MgPbO_3$, $MgSnO_3$, $MgZnO_3$, $SrAlO_3$, $SrPbO_3$, $SrZnO_3$
- **Alkaline earth metal / metalloid - cluster:** $BaGeO_3$, $BeGeO_3$, $BeSbO_3$, $CaSbO_3$, $SrGeO_3$, $SrSbO_3$
- **Lanthanide / transition metal - cluster:** $CeAgO_3$, $CeCoO_3$, $CeCrO_3$, $CeMnO_3$, $CeMoO_3$, $CeNbO_3$, $CeNiO_3$, $CePtO_3$, $CeRhO_3$, $CeRuO_3$, $CeTaO_3$, $CeTcO_3$, $CeTiO_3$, $CeVO_3$, $CeWO_3$, $CeZrO_3$, $LaAgO_3$, $LaCoO_3$, $LaCrO_3$, $LaCuO_3$, $LaFeO_3$, $LaMnO_3$, $LaMoO_3$, $LaNbO_3$, $LaNiO_3$, $LaPdO_3$, $LaPtO_3$, $LaRuO_3$, $LaTaO_3$, $LaTiO_3$, $LaVO_3$, $LaWO_3$, $LaZrO_3$, $NdAgO_3$, $NdCrO_3$, $NdMnO_3$, $NdMoO_3$, $NdNbO_3$, $NdNiO_3$, $NdPtO_3$, $NdRuO_3$, $NdTaO_3$, $NdTiO_3$, $NdVO_3$, $NdZrO_3$, $PmCrO_3$, $PmCuO_3$, $PmFeO_3$, $PmMnO_3$, $PmMoO_3$, $PmNbO_3$, $PmNiO_3$, $PmPdO_3$, $PmTaO_3$, $PmTcO_3$, $PmWO_3$, $PmZrO_3$, $PrAgO_3$, $PrCoO_3$, $PrCrO_3$, $PrFeO_3$, $PrMnO_3$, $PrNbO_3$, $PrNiO_3$, $PrPtO_3$, $PrRuO_3$, $PrTcO_3$, $PrTiO_3$, $PrVO_3$, $PrWO_3$, $SmAgO_3$, $SmCrO_3$, $SmFeO_3$, $SmNbO_3$, $SmNiO_3$, $SmPtO_3$, $SmRhO_3$, $SmRuO_3$, $SmTaO_3$, $SmTcO_3$, $SmTiO_3$, $SmVO_3$, $SmWO_3$, $SmZrO_3$
- **Lanthanide / other metal - cluster:** $CeBiO_3$, $CeCdO_3$, $CeGaO_3$, $CeSnO_3$, $LaAlO_3$, $LaGaO_3$, $LaSnO_3$, $LaZnO_3$, $NdAlO_3$, $NdBiO_3$, $NdCdO_3$, $NdGaO_3$, $NdPbO_3$, $NdSnO_3$, $PmAlO_3$, $PmBiO_3$, $PmCdO_3$, $PmGaO_3$, $PmPbO_3$, $PmSnO_3$, $PmZnO_3$, $PrAlO_3$, $PrBiO_3$, $PrGaO_3$, $PrPbO_3$, $PrSnO_3$, $PrZnO_3$, $SmAlO_3$, $SmBiO_3$, $SmCdO_3$, $SmGaO_3$, $SmSnO_3$
- **Lanthanide / metalloid - cluster:** $CeGeO_3$, $LaGeO_3$, $LaSbO_3$, $NdGeO_3$, $NdSbO_3$, $PmGeO_3$, $PmSbO_3$, $PrGeO_3$, $PrSbO_3$, $SmGeO_3$, $SmSbO_3$
- **Transition metal / transition metal - cluster:** $ScCoO_3$, $ScCrO_3$, $ScCuO_3$, $ScMnO_3$, $ScMoO_3$, $ScNbO_3$, $ScNiO_3$, $ScPdO_3$, $ScRhO_3$, $ScRuO_3$, $ScTcO_3$, $ScTiO_3$, $ScWO_3$, $ScZrO_3$, $YAgO_3$, $YCoO_3$, $YCrO_3$, $YCuO_3$, $YFeO_3$, $YMnO_3$,

$YMoO_3$, $YNiO_3$, $YPtO_3$, $YRhO_3$, $YRuO_3$, $YTaO_3$, $YTcO_3$, $YTiO_3$, YVO_3, YWO_3, $YZrO_3$

- **Transition metal / other metal - cluster:** $ScAlO_3$, $ScGaO_3$, $ScSnO_3$, $YAlO_3$, $YBiO_3$, $YCdO_3$, $YGaO_3$, $YPbO_3$, $YSnO_3$, $YZnO_3$
- **Transition metal / metalloid - cluster:** $ScSbO_3$, $YGeO_3$