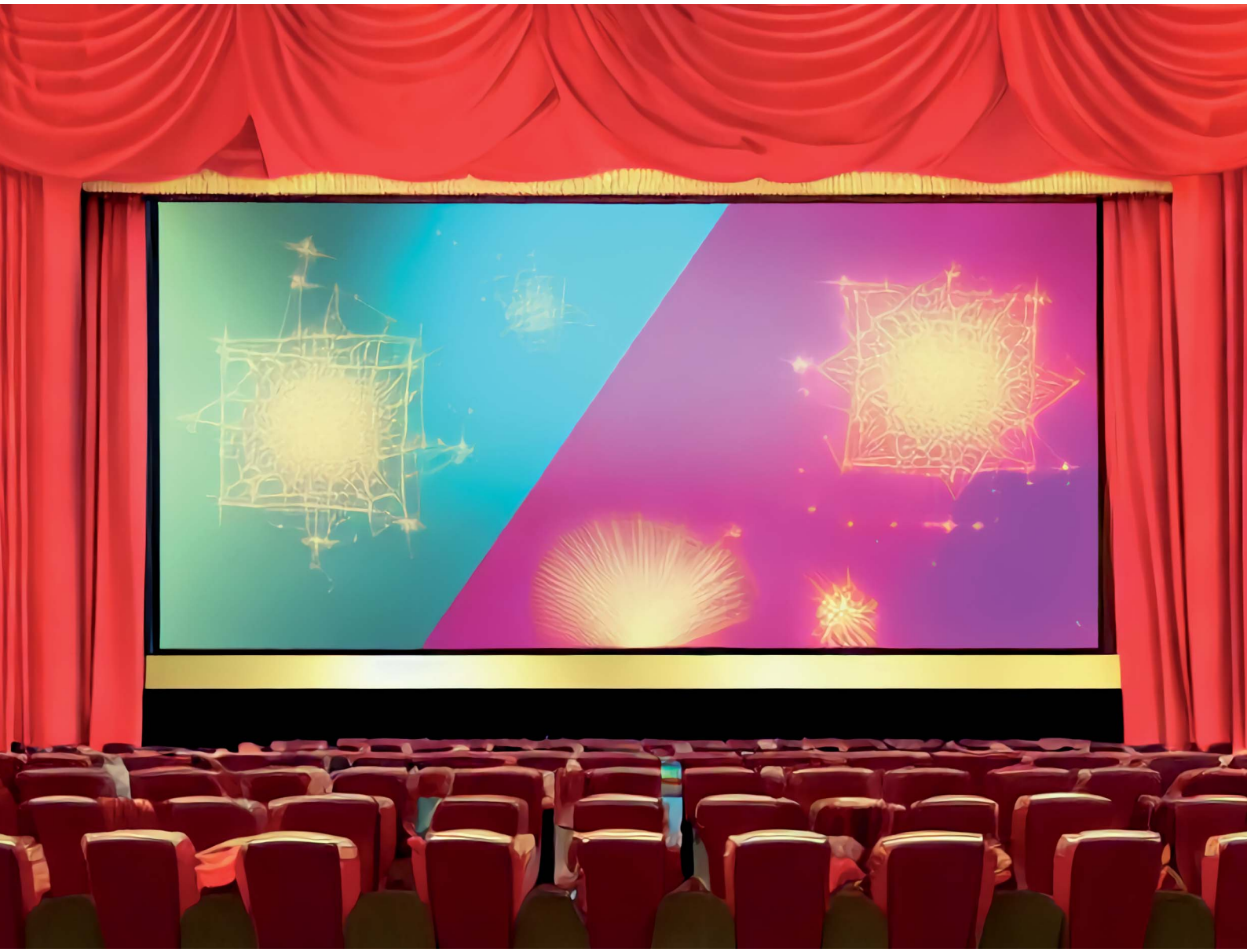
Volume 1
Number 6
December 2022
Pages 747-944

Digital Discovery

rsc.li/digitaldiscovery



ISSN 2635-098X

ROYAL SOCIETY OF CHEMISTRY

PAPER

Samantha Durdy *et al.*

Random projections and kernelised leave one cluster out cross validation: universal baselines and evaluation tools for supervised machine learning of material properties

Digital Discovery

ROYAL SOCIETY OF CHEMISTRY

PAPER

View Article Online
View Journal | View Issue

Cite this: *Digital Discovery*, 2022, **1**, 763

Random projections and kernelised leave one cluster out cross validation: universal baselines and evaluation tools for supervised machine learning of material properties†

Samantha Durdy, *[ab] Michael W. Gaultois,[bc] Vladimir V. Gusev,[bc] Danushka Bollegala[ab] and Matthew J. Rosseinsky[bc]

With machine learning being a popular topic in current computational materials science literature, creating representations for compounds has become common place. These representations are rarely compared, as evaluating their performance – and the performance of the algorithms that they are used with – is non-trivial. With many materials datasets containing bias and skew caused by the research process, leave one cluster out cross validation (LOCO-CV) has been introduced as a way of measuring the performance of an algorithm in predicting previously unseen groups of materials. This raises the question of the impact, and control, of the range of cluster sizes on the LOCO-CV measurement outcomes. We present a thorough comparison between composition-based representations, and investigate how kernel approximation functions can be used to better separate data to enhance LOCO-CV applications. We find that domain knowledge does not improve machine learning performance in most tasks tested, with band gap prediction being the notable exception. We also find that the radial basis function improves the linear separability of chemical datasets in all 10 datasets tested and provides a framework for the application of this function in the LOCO-CV process to improve the outcome of LOCO-CV measurements regardless of machine learning algorithm, choice of metric, and choice of compound representation. We recommend kernelised LOCO-CV as a training paradigm for those looking to measure the extrapolatory power of an algorithm on materials data.

Received 9th May 2022
Accepted 31st August 2022

DOI: 10.1039/d2dd00039c

rsc.li/digitaldiscovery

1 Introduction

Recent advances in materials science have seen a plethora of research into application of machine learning (ML) algorithms. Much of this research has focused on supervised ML methods, such as random forests (RFs) and neural networks. More recently, authors have laid out the best practices to help unify and progress this field.[1–4]

Data representation can play a large role in the performance of ML algorithms; however, optimum choice of representation is not always apparent. In materials science it is often difficult to choose an appropriate representation due to variability in the ML task and in the nature of the chemistry, composition and structures of the materials studied. Additionally, some properties of a material, such its crystal structure in the case of crystalline materials, may not be known until its synthesis. Accordingly, many studies derive representations from either the ratios of elements in the chemical composition, or from domain knowledge-based properties (referred to as features) of these elements, or both, in a process called "featurisation".

Given the ubiquity of featurisation methods such as those presented here in materials applications, it is important to evaluate the statistical advantage of specific feature sets.[5] Section 2.1 overviews different featurisation techniques and how their effectiveness has been previously reported. We expand on this evaluation in Section 3.1, in which seven representations are investigated across five case studies from the literature to explore how these representations perform in published ML tasks. These cases thus represent practical applications, rather than constructed tasks. Each of these representations is also compared to a random projection of equal size to establish the performance benefit of domain knowledge over random noise.

Evaluating the generalisability of ML models is a known challenge across data science, and is of particular concern in materials science, where data sets are of limited size compared with other application areas for ML, and often biased towards

[a]*Department of Computer Science, University of Liverpool, Ashton Street, Liverpool, L69 3BX, UK. E-mail: samantha.durdy@liverpool.ac.uk*

[b]*Leverhulme Research Centre for Functional Materials Design, University of Liverpool, 51 Oxford Street, Liverpool, L7 3NY, UK*

[c]*Department of Chemistry, University of Liverpool, Crown St, Liverpool, L69 7ZD, UK*

† Electronic supplementary information (ESI) available. See https://doi.org/10.1039/d2dd00039c

historically interesting materials or those closely related to known high-performance materials for certain performance metrics. Typically, models are evaluated on test sets separate from their training data, through a consistent train : test split or *N*-fold cross validation. However, this does not consider skew in a dataset. In chemical datasets, families of promising materials are often explored more thoroughly than the domain as a whole, which introduces bias and reduces the generalisability of ML models because the data they are trained and tested on are not sampled in a way representative of the domain of target chemistries to be screened with these models. Investigations into how such skew can affect ML models has seen that this skew can result in overfitting[6] and that more skewed datasets require more data points in order to train models to achieve similar predictive performance when compared to models trained on less skewed datasets.[7]

Leave one cluster out cross validation (LOCO-CV) was suggested to combat this,[8] using *K*-means clustering to exclude similar families of materials from the training set to measure the extrapolatory power of an ML algorithm (its ability to predict the performance of materials with chemistries qualitatively different from the training set). The value of such an approach can be seen in the case of predicting new classes of superconductors. One may choose to remove cuprate superconductors from the training set, and if an ML model can then successfully predict the existence of cuprate superconductors without prior knowledge of them, we can conclude that model is likely to perform better at predicting new classes of superconductors than a model which could not predict the existence of cuprate superconductors. LOCO-CV provides an algorithmic framework to measure the performance of models on predicting new classes of materials by defining these classes as clusters found by the *K*-means clustering algorithm. Application and implementation of this algorithm is discussed further in Section 2.2.1.

While differences in cluster sizes in this domain are expected, it has been observed that clusters found with *K*-means can differ in size by orders of magnitude,[9] which can pose a practical challenge to adoption of this method. With such differences in cluster size, LOCO-CV measurements can represent the performance of an algorithm on a small training set rather than the performance of an algorithm in extrapolation. As representation plays a role in clustering, it is pertinent to investigate the issues of representation and clustering together, even though the representation used in clustering does not need to be the same as that used to train the model (Fig. 3) In Section 3.2 we investigate how representations can affect measurements made with LOCO-CV. Kernel methods (also known as kernel tricks, or kernel approximation methods), can be used to non-linearly translate data into a data space that can then be linearly separated (Fig. 2). We apply kernel methods such as the radial basis function (RBF) to chemical datasets to improve the linear separability of data and reduce variance between cluster sizes and thus increase the validity of LOCO-CV measurements (Fig. 5 and 8), thus enhancing the assessment of performance found when using different representations as well as assessment of model performance as a whole.

LOCO-CV evaluation is affected by representation of a compound and, conversely, choice of compound representation is affected by the methods used to evaluate these representations. Thus, it is pertinent to investigate these two issues simultaneously. We improve the utility of LOCO-CV measurements by using kernel functions to create a more separable data space, and use these measurements to evaluate featurisation methods using practical supervised ML tasks found in the literature. The key contributions and findings of this paper are as follows:

- Comparing the influence of composition based feature vectors (CBFVs) on ML model performance in practical tasks (explained further in Section 2.1, before being carried out in Section 3.1). We find that CBFVs with engineered features (*i.e.*, imbued with domain knowledge) do see some benefit in certain tasks, particularly band gap prediction tasks. While magpie representations[10] were seen to outperform other CBFVs in many tasks, this finding was not universal across tasks.
- Examining the effectiveness of random projections as featurisation methods for property prediction from chemical composition. Random projections can be used as a baseline against which to justify more involved featurisation methods (explained further in Section 2.1.2 before being carried out in Section 3.1). We find that in many tasks, CBFVs with engineered features do not perform substantially better than random projections.
- Studying the effect of kernel approximation functions (explained further in Section 2.3) on the application of *K*-means clustering to materials data, and presenting a workflow to incorporate these methods into the LOCO-CV algorithm (Section 3.2). We find kernel approximation functions are a good way to reduce the variance between sizes of clusters found by *K*-means clustering on materials data. Using kernel approximation functions in the suggested workflow (kernelised LOCO-CV) results in a more robust evaluation method than LOCO-CV with no kernels.
- We recommend using RBF when clustering for LOCO-CV, as clusterings found after application of RBF are seen to be more even in size than with no kernel method applied, and models are trained more reliably for property prediction. This helps to reduce the risk that performance differences on predicting an unseen cluster of data are caused by the training set size as opposed to the intrinsic inability of a model to perform well on that cluster of data.
- We find the use of the radial basis function (RBF) in clustering for LOCO-CV leads to more reliable and consistent model training, compared to using LOCO-CV without any kernel methods.
- We recommend that random projections are used as a baseline against which to compare engineered feature vectors, noting that commonly used CBFVs have little to no advantage over random projections in most tasks tested here.
- We experiment with the use of random projections as a featurisation method for clustering compositions in LOCO-CV, and find random projections to have no clear advantage over other CBFVs tested here.

2 Key concepts and techniques

2.1 Common representations used for machine learning in inorganic chemistry

ML algorithms require a consistent definition of a data point in order to analyse trends within a dataset. For example, it would be hard to learn from a dataset in which "a data point" may refer to a phase field, a specific crystal structure, or a composition. One such algorithm is RFs, which are widely used in materials science as well as other domains.[11] They are fast to train, readily implemented,[12] and see a good performance in a plethora of tasks without hyperparameter tuning. We use RFs for our investigations for reasons outlined above, however good evaluation methods for fixed dimensional representations of materials are also important for the plethora of other ML algorithms that use such representations as basis for predictions.

Representation learning, and feature engineering are the two main preprocessing methods to make data more interpretable to ML algorithms. Representation learning is a fast-evolving field that uses deep learning in order to create representations, while feature engineering involves defining a set of features (or descriptors) for a data point that adequately encapsulates all information needed.[13]

Feature engineering has been used extensively in inorganic chemistry and materials science. However, no set of features has emerged as the clearly dominant representation for a material, likely due to the variety of tasks carried out in these domains, which may require different input representation. Many of these representations use only composition-based information (rather than structural), as this allows screening of materials without need for DFT calculations or synthesis, greatly reducing costs associated with such screenings. Composition-based screening is less powerful than the incorporation of structure, as both structure and composition control properties, but more general as structural information is not required and is less widely available than composition (as structure is not known until the material is realised by synthesis, whereas compositions can be proposed without knowing structure). Composition-based feature vectors (CBFVs), which offer a list of compositional attributes of a material, and a one-hot style (also called fractional) encoding of composition,[14] are widely used composition-based representations.

Notable CBFVs including magpie, Oliynyk and JARVIS[15–17] (differences between which are discussed further during Section 3.1) were recently investigated and found to provide benefit over one-hot style representations. This benefit was measured using neural networks predicting numerous properties, however the benefit became little to none as the dataset size increased above 1000 points.[5]

We further the investigation into the use of CBFVs by examining their applicability in five case studies. Namely, we examine performance using Oliynyk, magpie, and JARVIS, a variant of random projection of size 200 (discussed more in Section 2.1.2) used in a previous review on this topic,[5] as well as one-hot style encodings of composition, and random linear projection of the composition. The performance of RFs using different representations are compared on ML tasks found in the literature, using the relevant datasets for each study.[18–22]

The representations were chosen as they are commonly used, and as these are the non-structural representations investigated for their efficacy in neural networks in previous work.[5] Seeing whether previous results hold for RFs should help gauge whether these results could be used as rule of thumb for many ML algorithms or whether these conclusions should only be applied to neural networks similar to those used in that study.

2.1.1 Can implementation details in CBFVs affect performance. It is common for a CBFV to be comprised of a list of elemental properties that are combined using several "aggregation functions", for example the weighted average, and standard deviation of various elemental properties in a compound (Fig. 1a). The aggregation functions of a CBFV can vary between implementations.[5,15] Using different numbers of aggregation functions results in representations of different lengths (Fig. 1a), which may affect ML performance depending on the algorithm being used.

Problems associated with building statistical models using increasingly large data representations without also increasing the number of data points are well documented, often being described as the curse of dimensionality.[23] Strong correlation between different dimensions (known as co-linearity, or cross correlation between dimensions) can also impact model performance. For example, RFs are affected by co-linearity between dimensions as RF's random bagging process is unlikely to select a subset of features that include none of a set of cross corelated features. This would make the information in features with such cross-corelates more likely to be available to discriminate with at any branch in a tree, compared with those features without such cross-corelates. It is intuitive that different aggregation functions may be cross-correlated, for example the maximum atomic weight of an element in a compound is likely to correlate with the average atomic weight of an element in that compound, thus RFs may be affected by additional aggregation functions.

Without investigation, it is unclear what effect different aggregation functions will have on algorithm performance. Interrogation of the repository associated with the previous review of featurisation methods indicates use of the weighted average, sum, range, and variance of each feature.[5] This includes the features of the fractional (one-hot style) representation, which uses only the ratios of each element in a material in its definition. This implementation difference could affect the performance of a model that uses these representations, so we distinguish between the two, using "fractional" to refer to a one-hot style encoding that includes the average, sum, range, and variance of each element and "CompVec" (for composition vector) to refer to an implementation of one-hot style encoding which contains just the ratios of elements in a compound.

The nature of the fractional representation means that a given compound would contain the same representation three

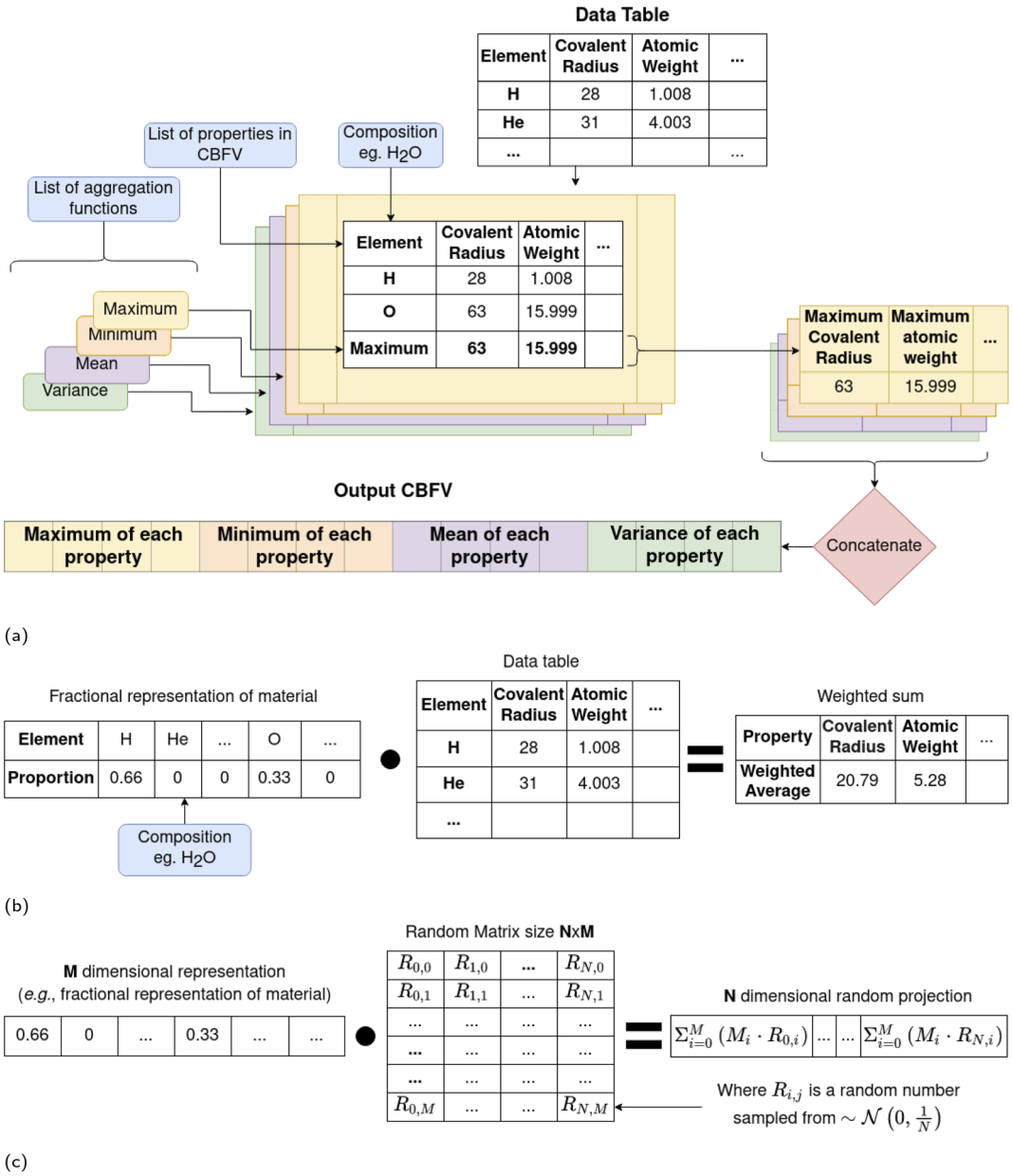


Fig. 1 Comparison of the creation of composition based feature vectors (CBFVs) and random projections. (a) General workflow for creation of CBFV. Application of aggregation function to each property of a material will result in a fixed sized vector for each aggregation function, these are then concatenated together (merged sequentially) to form the final CBFV. Both the properties in the CBFV and the list of aggregation functions can be changed to create variants of CBFVs, which may influence algorithms that use the resulting CBFV. (b) Calculation of the weighted sum of properties of a material. This is equivalent to the matrix multiplication of the fractional representation of that material and its properties. (c) Calculation of a random projection. Using random projection to (approximately) linearly project a representation into a different number of dimensions (N). The original M dimensional representation for our purposes may be a fractional representation for the chemical composition of a material, but this technique can be used for any input data, in domains outside of chemistry.

times, scaled by different amounts (depending on the number of elements in the compound) in a single vector (four times if elements in a compound are in equal ratios). This can be exemplified by examining a simple composition such as NaCl (Table 1).

This offers an opportunity to investigate how increasing dimensionality (the number of dimensions) of a representation while adding no new information affects performance. We leave the investigation of the effect of information added by different aggregation functions on different feature sets to future work. We experiment using both a (CompVec) one-hot style encoding as proposed for use with ElemNet[14] (with no additional aggregation functions), and the one-hot style approach used previously that includes different aggregation functions (fractional),[5] to see how this increase in dimensionality above will affect experiments.

Table 1 Values that would occur in each column across different aggregation functions for a composition fractional representation of NaCl. This demonstrates how the inclusion of additional aggregation functions does not add additional information for this representation. These calculations assume a representation which allows for 118 different elements, a smaller number of represented elements would result in the values in the variance columns being larger

Aggregation function	Na	Cl	All other columns
Weighted average	0.5	0.5	0
Sum	1	1	0
Range	1	1	0
Variance	0.0042	0.0042	0

While this increase in dimensionality will be seen to affect the clusterings found with *K*-means clusterings, for most tasks investigated there was not an appreciable difference between CompVec and fractional representations. In band gap prediction tasks fractional representation outperformed CompVec, however in regression tasks relating to bulk metallic glass formation this trend was reversed (Fig. 4).

2.1.2 Random vectors as featurisation methods. Each elemental property (for example covalent radius) aims to bring with it some sort of information about that element. That property's inclusion in a feature set aims to improve an ML algorithm's performance in a given problem. Every feature included either means an increase to the dimensionality of a CBFV or the exclusion of an alternative feature. Though the importance of a feature to an ML model can be measured,[24,25] it is hard to take such measures of feature importance out of the context of the model that is trained with it, or the dataset that the model is derived from ref. 26.

As it is hard to distinguish the effects of dimensionality of a representation from the effects of the information imbued in it, Murdock *et al.* introduce a set of vectors, one for each element each consisting of 200 random numbers to represent nonsensical elemental properties. From these vectors, they derive the CBFV RANDOM_200 to represent a lower bound for feature performance. That is to say; rather than using features that would be expected to give information about an element (covalent radius, atomic number *etc.*), they instead assign each element a vector of random numbers. If these random numbers can result in a well-performing model then whether the chemically-derived features that are commonplace in the literature are justified can be called into question. When the aggregation function is a weighted sum (discussed further in Section 2.1.1), this has the same effect as a matrix multiplication of the one-hot style encoding of a compounds formulae, C, (referred to in this paper as CompVec), and a random matrix, R which can be noted as $C \cdot R$ (Fig. 1b). Thus the weighted sum part of the RANDOM_200 can be seen as a matrix multiplication of the random vectors and the fractional encoding of the composition.

This matrix multiplication is similar to that used in a random projection. Random projection is a dimensionality reduction technique that uses the observation that in high dimensions random vectors approach orthogonality.[27,28] When the columns of R are normalised to be unit vectors, $C \cdot R$ becomes an approximately linear projection of C. Another way to closely approximate normalisation of the columns of a random matrix, such as R, is to sample the values of that matrix from a Gaussian distribution of mean 0 and variance $\frac{1}{N}$ $\left(\sim\mathcal{N}\left(0,\ \frac{1}{N}\right)\right)$ where N is the size of the projection. This is mathematically justified by the Johnson–Lindenstrauss lemma, which states that for a set of N dimensional data points there exists a linear mapping that will embed these points into an n dimensional data space while preserving distances between data points within some error value, ε. This value of ε is shown to decrease as n increases[29]

RANDOM_200 samples from $\sim$(0,1) also included aggregation functions (namely sum, range, and variance),[5] as discussed in Section 2.1.1. It is unclear what impact this will have however preliminary investigations show little difference in performance between sampling from $\sim$(0,1) and $\sim\mathcal{N}\left(0,\ \frac{1}{N}\right)$.

We investigate the use of random projection as an alternative to more widely used techniques by comparing each technique investigated to a random projection of the same size (Fig. 4). This should allow us to note improvements made by the quality of features as opposed to the quantity. We include RANDOM_200 in this investigation, noting the key difference between this and the random projection being that the random numbers are drawn from different distributions (as outlined above) and that RANDOM_200 includes aggregation functions, where a random projection does not.

2.2 Training methods for materials science

Performance metrics are usually applied to a test set of data unseen by a model. Where data are scarcer, or computation time is not limiting, *N*-fold cross validation can be used. This is often referred to as *K*-fold cross validation but we use *N* to avoid confusion with *K*-means clustering, a more central algorithm to this work. *N*-fold cross validation randomly splits data into *N* equal sized random "folds", *N* models are then trained, each model trained on all but one of the folds of data, and evaluated on the fold which is held out. Performance is then averaged. A common criticism of supervised ML in materials science is that datasets being worked with are inherently biased. Bias in data is a problem more broadly in ML research. In this field, exploration of similar, promising chemistries for particular applications leads to areas of the chemical data space being more dense with successfully synthesised (or DFT calculated) materials than others.

This leads to inflated performance metrics as performance can only be measured against other compounds that have already been synthesised (or compounds with relevant DFT calculations), as many such compounds in the test set will have similar chemistry to the training set. This can lead to comparatively poor results when trying to extrapolate to predict properties for chemistries dissimilar to those that the algorithm has been trained on. For example, it could be argued that the entries

in ICSD reflect a bias towards the development of both analogues of the chemistry of minerals and chemistries lending themselves to specific types of application performance, rather than an isotropic exploration of chemical space constrained only by the inorganic chemistry of the elements themselves. Such considerations emphasise the importance of discovery synthesis that accesses new regions of chemical space, as the resulting materials can contribute to more robust models. Having robust methods to measure model performance is pertinent for materials discovery to assess likely model effectiveness in extrapolating to unseen areas of the input domain.

2.2.1 Leave one cluster out cross validation (LOCO-CV). A method to measure the extrapolatory power of an algorithm was proposed in leave one cluster out cross validation (LOCO-CV).[8] LOCO-CV alters *N*-fold cross validation to have each fold contain materials in the same cluster rather than randomly selected (equal sized) folds, in order to emulate performance on unseen classes of materials.

Clusterings are selected using the *K*-means clustering algorithm,[30,31] which infers *K* clusters without the need for target labels. This is done by grouping data into clusters based on their Euclidean distance to K randomly chosen "centroids". The centroids are then redefined as the mean of all points in a cluster and the data are regrouped based on these new centroids. This process is repeated until the positions of centroids (or the contents of their associated clusters) converge. *K*-means is quick, robust and readily implemented.[12]

One concern often raised with LOCO-CV is how its non-deterministic nature will affect the repeatability of measurements taken using this evaluation method. In the ESI (Section S3.2†) we outline experiments performed to test how repeatable LOCO-CV is, finding that while it is less repeatable than using an 80 : 20 train:test split to evaluate a random forest, it is the deviation between measurements made were not sufficient to substantially impact the interpretation of the results seen in this paper.

LOCO-CV as explored here uses *K*-means clustering with values of *K* between 2 and 10 (inclusive), taking the mean of the resulting metrics. This is the version of LOCO-CV most thoroughly explored by the authors of LOCO-CV (though they use the median rather than the mean). However, alternative methods of selecting a single value of *K* were suggested in that work. Namely alternatives suggested were use of X-means,[32] G-means,[33] or silhouette factor threshold[34] for selection of *K*.

LOCO-CV does however leave representation as a hyperparameter to the clustering (*i.e.*, changing the representation will change the clusterings found with *K*-means clustering), and that the stochastic nature of the *K*-means algorithm can make measurements hard to reproduce without publishing the clusters found. A further consideration in use of LOCO-CV is that *K*-means does not guarantee the size of any clusters, nor does it guarantee that clusters would be deemed chemically sensible (this is discussed further in Section 2.4). It has been observed that clusters taken on materials data can vary in size by multiple orders of magnitude, which hinders the application of LOCO-CV.[9]

While different sizes of clusters are to be expected in this domain (for example due to research bias in the generation of example materials), should the sizes of the clusters found in LOCO-CV differ by orders of magnitude then LOCO-CV's ability to measure extrapolatory power is hampered. Intuitively if one of ten clusters contains 90% of the materials in the dataset, then a measurement made with this cluster left out may give a measurement of algorithmic performance given a small fraction of the available training data, rather than indicating extrapolatory power. *K*-means clustering by its nature can only linearly separate clusters in a given data space. Clusters that are more distinct from one another are more likely to be isolated than clusters of data points that overlap with each other. There are other clustering algorithms, such as *t*-distributed stochastic neighbour embedding,[35] agglomerative clustering,[36] or DBSCAN,[37] that could be explored for LOCO-CV applications on materials datasets. We measure the separability of clusters of compounds in materials science datasets with *K*-means clustering.

2.3 Kernel methods

While uneven cluster sizes do pose problems for LOCO-CV assessment of the extrapolatory power of ML models, such issues with *K*-means clustering are not solely found in materials science. *K*-means clustering attempts to linearly separate clusters (*i.e.* draw a straight line between them), some clusters cannot be separated this way (Fig. 2). In many cases, applying a non-linear function to every point in the dataset transforms the data in such a way that clusters can be linearly separated.[38] Functions used to preprocess data in this way are called kernel methods (also kernel approximation methods or kernel tricks). Prominent examples of this include RBF,[38] additive χ^2,[39] and skewed χ^2.[39] We look at the first of these in more detail to illustrate how such kernel methods affect data points. The RBF can be defined as:

$$f(x) = \exp(-\gamma x^2)$$

where γ is a hyperparameter which was set as 1 throughout this study (1 is the default for this hyperparameter in the library used). Here $x \in D$ where D is a dataset of materials each represented by a feature set R^n where n is the dimensionality of the feature set. Examination of this formula lends intuition to the effects seen in its application (Fig. 2), but also highlights that this function does distort the geometry of an input data space. Thus, some analysis of the results of this function are inappropriate, such as inferring meaning from changes in distances between specific points. Despite these potential caveats, non-linear transformations (*e.g.*, through application of kernels) are frequently used with linear discrimination (such as *K*-means clustering).[38] In this paper we investigate the effect of kernel methods such as RBF on materials science data, specifically studying the use of such methods to improve suitability of LOCO-CV by addressing the problem of uneven cluster sizes outlined in Section 2.2.1. We find RBFs reduce the variance of class sizes in a clustering, regardless of input featurisation and

note that this results in more reliable model training when using these clusterings for LOCO-CV.

2.4 Performance metrics in *K*-means clustering

Without prior knowledge of expected clusters for each data point, results found with *K*-means clustering are difficult to interpret, though expert inspection can yield insights into what different clusters can represent. Expert inspection of results may be justifiable with less than 10 clusters (each of which could have thousands of materials), however, when using K between 2 and 10 (as was originally proposed[8]), the LOCO-CV algorithm presents 54 different clusters ($\sum_{n=2}^{10} n$), making such expert inspection infeasible. Thus metrics must be used to quantify the success of a clustering.

Where target labels exist, metrics such as mutual information score, homogeneity, and completeness scores can be used. Without labels, Euclidean distance-based measures such as sum-squared distance to cluster centroid or average distance between each point and the other points in its cluster can be used, however this does not intrinsically tell us how much information is in a clustering, just how tightly packed a cluster's members are. The average distance between each point and the other points in its cluster is computationally prohibitive so will not be used in this study.

Euclidean distance-based measurements such as these lack comparability in our use case, as each dataset and each featurisation technique should be considered independent. Identifying trends in these measurements with different numbers of clusters and looking at the effect of kernel methods on Euclidean distance-based measurements are both valid uses. However, as Euclidean space is affected by dimensionality, it is important that conclusions into the effect of different featurisation approaches are not drawn from such measures. While noting these caveats, we use the mean distance of a point in a cluster to the cluster's centroid as a measure of how tight the clusters are in Euclidean space, we label this metric the spread of cluster.

As the aim of this investigation is to improve the validity of measures taken with LOCO-CV, specifically to address issues with vastly uneven cluster sizes, we also use the standard deviation in cluster sizes as a metric for success (the unevenness in cluster sizes). Material science datasets may have uneven cluster sizes due to research bias towards exploration of promising materials, and identically sized clusters would be unexpected for materials data, identically sized clusters were, in practice, never observed in this study. Using the unevenness of cluster sizes serves as a measure of whether cluster sizes differ by many orders of magnitude, which would affect the validity of measurements taken using LOCO-CV. This does not imply that more even clusters are more chemically sensible groupings of materials, just that they may be more sensible for use with LOCO-CV, as uneven cluster sizes bring into question measurements taken with LOCO-CV (Section 2.2.1).

The ease of clustering is expected to vary between datasets. Accordingly, to appropriately to compare standard deviation in cluster sizes, we perform max–min normalisation across different featurisation techniques and numbers of clusters in the same dataset. Consequently, for each dataset, the most uneven cluster size measurement found is 1 and the least uneven cluster size measurement is 0. We use these normalised values when comparing cluster size unevenness between datasets.

3 Results

3.1 Effect of representation on predictive ability of random forest: case studies

We examine five case study publications' datasets to compare the representations used in them with a non-structural CBFV examined in previous work,[5] and with the composition vector (CompVec) suggested for use with ElemNet.[14] Case studies have been selected to incorporate the prediction of a variety of material properties, research groups, and notable works that

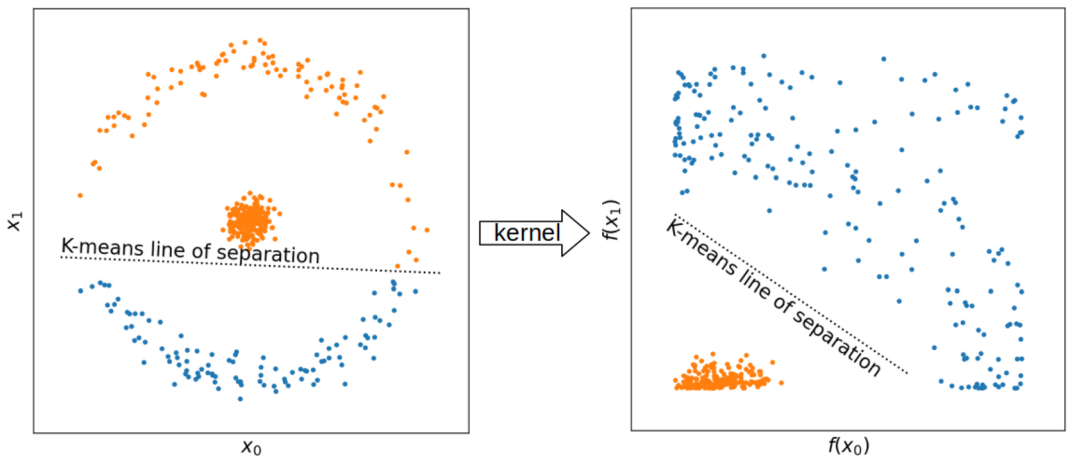


Fig. 2 A visualisation of how application of kernel functions can affect the data in an example dataset. Here we show the radial basis function (RBF) so $f(x) = \exp(-x^2)$. There is no clear way to linearly separate classes before application of RBF; however, non linear translation of each point with the RBF yields a data space through which a straight line can be drawn to separate the classes.

reflect the state-of-the-art. We use the original datasets to replicate studies, but use 80 : 20 train:test splits.

We use a consistent 80 : 20 train:test split across all data sets to enable us to draw conclusions about which representations work better generally. This should help us to establish whether previous findings (*i.e.* that domain knowledge is more beneficial in smaller datasets and that benefit diminishes as dataset size increases over 1000),[5] hold true for RFs. LOCO-CV measurements for these experiments are available in the ESI,† and the clusterings found for LOCO-CV are available in the associated git repository.[40]

Representations compared are:

- Oliynyk.[16] Originally designed for prediction of Heusler structured intermetallics,[16] the Oliynyk feature set as implemented in previous work includes 44 features.[5] For each of these, the weighted mean, sum, range, and variance of that feature amongst the constituent elements of the compound are taken. Features include atomic weight, metal, metalloid or non metallic properties, periodic table based properties (Period, group, atomic number), various measures of radii (atomic, Miracle, covalent), electronegativity, valency features (such as the number of s, p, d, and f valence electrons), and thermal features (such as boiling point and specific heat capacity).
- JARVIS:[17] JARVIS combines structural descriptors with chemical descriptors to create "classical force-field inspired descriptors" (CFID). Structural descriptors include bond angle distributions neighbouring atomic sites, dihedral atom distributions, and radial distributions, among others. Chemical descriptors used include atomic mass, and mean charge distributions. Original work generated CFIDs for tens of thousands of DFT-calculated crystal structures,[17] and subsequent work adapted CFIDs for individual elements to be used in CBFVs for arbitrary compositions without known structures (*i.e.* Fig. 1a).[5]
- magpie:[15] While the Materials-Agnostic Platform for Informatics and Exploration (MAGPIE) is the name of a library associated with Ward *et al.*'s work, it this has become synonymous with the 115 features used in the paper and, as such, we

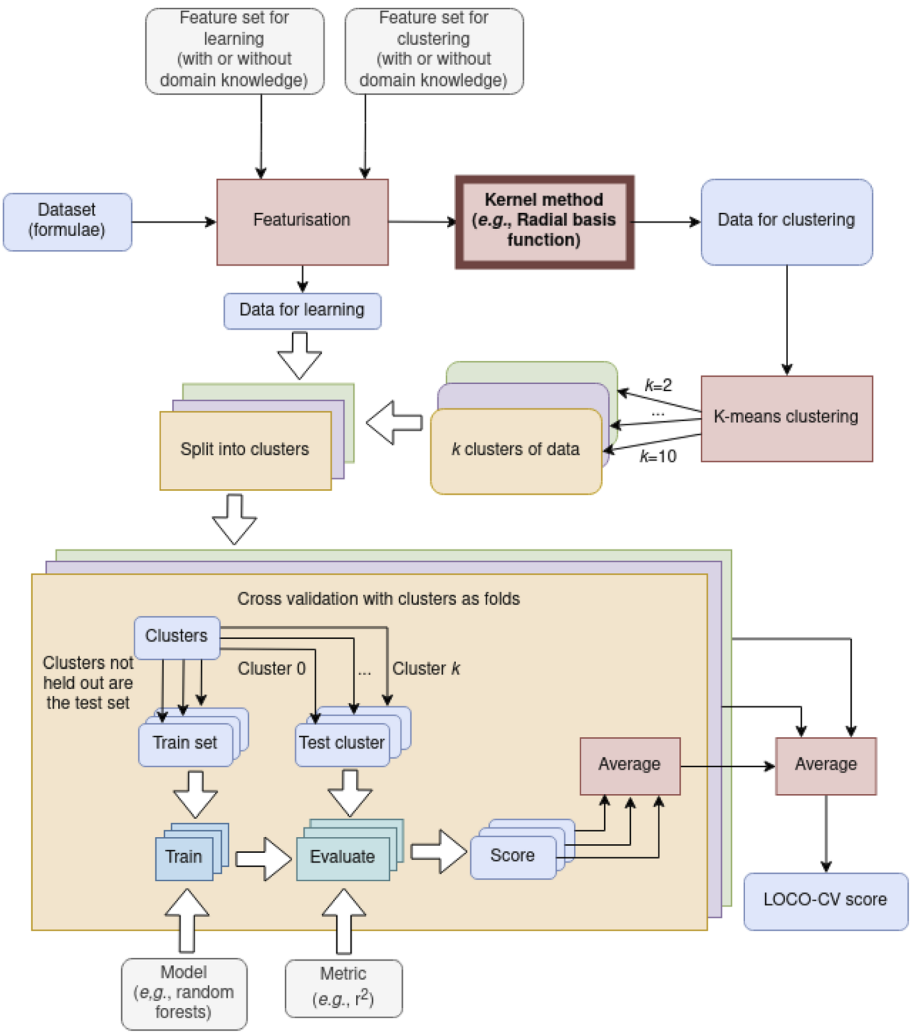


Fig. 3 A flow chart of the kernelised LOCO-CV process in a property prediction task. The novel kernel application is highlighted in a bold frame. Note that the representation used for clustering is independent of that used for training the models. Consequently, kernel methods can be easily integrated into existing property prediction workflows without changes to how models are trained.

will use magpie refer to the feature set. These features include 6 stoichiometric attributes which are different normalisation methods (L^P norms) of the elements present. These capture information of the ratios of the elements in a material without taking into account what the elements are, 115 elemental based attributes are used, which are derived from the minimum, maximum, range, standard deviation, mode (property of the most prevalent element) and weighted average of 23 elemental properties including atomic number, Mendeleev number, atomic weight among others. Remaining features are derived from valence orbital occupation, and ionic compound attributes (which are based on differences between electronegativity between constituent elements in a compound).

- RANDOM_200:[5] a random vector featurisation used by Murdock *et al.* to represent a lower bound for performance.
- Fractional:[5] An implementation of a one-hot style encoding of composition which includes average, sum, range, and variance of each element.
- CompVec a one-hot style encoding of composition as used in ElemNet[14] (containing only the proportions of each element in a composition). Differences between this and fractional are further discussed in Section 2.1.

We compare each of these representations to a random projection of equal size. This allows us to control for the size of a representation when investigating the advantage of the domain knowledge built into a CBFV. Several of the five case studies investigated contain multiple applications of ML within a single publication. The tasks which were recreated in this comparison (and their relevant case study references) are as follows:

- T_c: using a regressor to predict the superconducting critical temperature (T_c) of a material (12 666 data points in training set).[18]
- $T_c > 10$ K: classifying if the T_c of a material is greater than 10 K (12 666 data points in training set).[18]
- $T_c|(T_c > 10\text{ K})$: regressing to find T_c given $T_c > 10$ K (4833 data points in training set).[18]
- HH stability: predicting the stability of half-Heuslers (8948 data points in training set).[19]
- E_{gap}(oxides): predicting the band gap of oxides found in the Computational Materials Repository database (599 data points in training set).[21]
- Glass Forming Ability (GFA): predicting the ability of a bulk metallic glass alloy (BMG) to exist in an amorphous state (5051 data points in training set).[20]
- D_{max}: predicting the critical casting diameter of a BMG (4724 data points in training set).[20]
- ΔT_x: the supercooled liquid range of a BMG (495 data points in training set).[20]
- E_{gap}(DFT): predicting the band gap of materials calculated using DFT (35 653 data points in training set).[22] This dataset combines data from the materials project and Duke University's AFLOW.[41,42]
- E_{gap}(exptl): predicting the band gap of materials measured experimentally (1986 data points in training set).[43] This was used in experiments as to the effect of transfer learning from DFT to experimental band gap prediction.[22]
- E_{gap}(DFT)$\cup E_{gap}$(exptl): predicting the band gap of a dataset consisting of both DFT calculated and experimentally measured band gaps (37 639 data points in training set).[22]

We report measured performance in regression tasks was using r^2 correlation and classification task performance is measured using accuracy. Thus percentage improvement over random projections can be considered to be:

$$100\left(\frac{M(y,\hat{y})}{M(y,\hat{y}_p)}-1\right)$$

Where y is the target label for a prediction, $\hat{y}$ is the label predicted by a model that uses a given representation, $\hat{y}_p$ is a label predicted by a model that uses a random projection of equal size to the given representation, and M is accuracy for classification tasks and r^2 for regression tasks. Measurements found using other values of M can be found in the ESI.† To investigate repeatability of these results, a large subset of these experiments have been repeated 5 times and the standard deviations of these results calculated. This can be found in the ESI.†

Overall, recreation of these tasks shows that, broadly, changes in CBFV made little difference to performance when compared to a random projection of the same size (Fig. 4). Featurisation methods inspired by domain knowledge do show advantages in some datasets. These advantages seem to be task-specific as opposed to based on dataset size, specifically band gap-based tasks seem to see benefit from knowledge-based features, however most other tasks do not see noticeable improvement from this feature engineering (Fig. 4). This could be because vast amounts of band gap data can be acquired through DFT calculations[41] and as such band gap prediction is a widely available benchmark that researchers could use when testing a newly proposed CBFV.[44]

Intuition may suggest introducing more dimensions that do not contain any additional information would result in worse

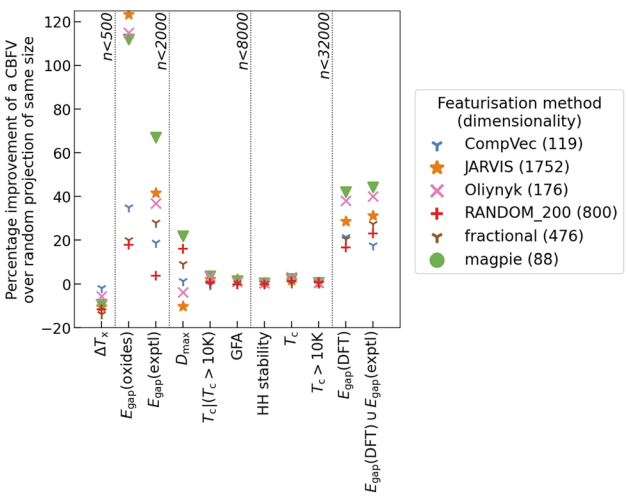


Fig. 4 Performance of composition-based feature vectors (CBFVs) on predictive tasks compared to random projections. Random projections exhibit similar performance to CBFVs for most tasks. This is not true for band gap prediction tasks, where CBFVs with domain knowledge demonstrate marked improvement.

algorithmic performance. However, despite having 68% more dimensions, RANDOM_200 performs within 5% of the fractional representation. On large enough data sets ($\sim 3000 < n$) the random representation does not perform appreciably differently to the magpie representation. Notably on tasks outside of band gap prediction there is little advantage to domain based representations over a random projection.

We encourage the use of random projection as an alternative to CBFV, and propose its use as a comparative measure against CBFV. If a feature set cannot appreciably outperform a random projection of the same size or smaller, then, while there may still be benefits to analysis of the feature importance of such a feature set, that feature set does not enrich the representation of a material when it comes to algorithmic performance.

3.2 Improving the linear separability of chemical data spaces for more applicable measurements of extrapolatory power

We investigated which of the representations of a compound outlined in Section 3.1 will lead *K*-means clustering to identify more evenly sized clusters in different datasets. Datasets investigated were those used in Section 3.1 as well as the inorganic crystal structures database (ICSD) as a whole.

In classical computer science problems, non-linear kernels have been applied to datasets on which a linear discriminator (such as *K*-means, or support vector machines) exhibits poor performance. As described in 2.3, applying a non-linear transformation (*e.g.*, a kernel function) to every data point in a data set can transform data such that it is more amenable to linear discrimination (Fig. 2). We applied the radial basis, additive $\tilde{\chi}^2$, and skewed $\tilde{\chi}^2$ functions to the investigated representations to see if these non-linear translations will reduce cluster size unevenness found by *K*-means clustering. Reduced cluster size unevenness found with *K*-means would improve the applicability of LOCO-CV measurements, addressing one of the problems highlighted in Section 2.2.

As additive $\tilde{\chi}^2$, and skewed $\tilde{\chi}^2$ functions are only well defined for positive inputs, data was scaled between 0 and 1 using min–max normalisation before these methods were applied. As RBF (and *K*-means without kernels) can be affected by disparity of scale between axes, different normalisation methods were investigated, with the data normalisation which most often resulted in the lowest cluster size unevenness being used for the results below (no normalisation was used with RBF and min–max scaling to between −1 and 1 was used when no kernel method was being applied). Further details of this can be seen in Section S1 of the ESI.†

All three kernel functions investigated resulted in more evenly sized clusters than no kernel function being applied at all, with RBF, on average, resulting in the largest reduction in standard deviation between cluster size (Fig. 5). Additionally, we note that application of any of these kernel methods generally resulted in a reduction in distance between points in a cluster and their centroids (spread of cluster), indicating more tightly packed clusters (Fig. 6b). On average application of skewed $\tilde{\chi}^2$ saw the greatest reduction in spread of cluster. As this investigation looks to create more even cluster sizes for use with LOCO-CV we focus on impacts of RBF, as, of the kernel methods tested, it resulted in the greatest impact on this metric as defined by the largest reduction in standard deviation of cluster size.

Before application of a kernel function, we note that cluster sizes are more even in domain knowledge-based representations as measured by the standard deviation in cluster sizes. CompVec representation resulted in a larger standard deviation between cluster sizes (*i.e.*, less evenly sized clusters) than all other representations investigated, likely due to the sparse nature of this representation, with the magpie representation resulting in the most even cluster sizes (Fig. 7a). The two one-hot based representations, fractional and CompVec, generally did not result in as even cluster sizes as other representations. Application of CompVec resulted in performance substantially worse than that of fractional despite them being very similar nature, only differing in use of aggregation functions (as discussed in Section 2.1).

RBF universally resulted in more even clusters. The smallest change (as a percentage of the standard deviation in cluster size before application of RBF), was seen in fractional and CompVec representations (two of the representations which resulted in the worst performance in this metric) (Fig. 6a). However, outside these two representations, the proportional impact of RBF on this measure did not correlate to the performance of a CBFV in this measure prior to application of RBF.

Without use of kernel functions, there is a clear correlation between the size of a representation and the spread of the clusters found using that representation, with the exception of CompVec, which saw the tightest clusters (Fig. 7b). This trend is no longer seen after application of RBF. Application of RBF to a CBFV before *K*-means clustering reduced the spread of clusters found (Fig. 6b and 7b). The relative size of the change seen after application of RBF correlated with the spread of clusters found when no kernel method was used. The higher the spread of clusters found using a CBFV without a kernel method, the larger the change seen when clustering using that CBFV and a RBF.

Use of kernel methods in featurisation results in more even cluster sizes when using that featurisation for *K*-means clustering. As featurisation used for clustering in LOCO-CV is independent of that used for learning, incorporating these kernel methods into LOCO-CV is simple and applicable regardless of machine learning algorithm, chosen metric, and initial representation (Fig. 3). Thus we recommend use of kernel methods when using *K*-means clustering for LOCO-CV to address the issue of uneven cluster sizes (as discussed in Section 2.2). Addressing this issue results in models being more reliably successful at learning trends in data using LOCO-CV (Fig. 8).

3.3 Clustering random projections with and without kernel methods

Having established that random projections perform similarly to engineered feature vectors in many task (Section 3.1) and that kernel methods can be used to reduce cluster size variance in *K*-means clustering on materials datasets (Section 3.2),

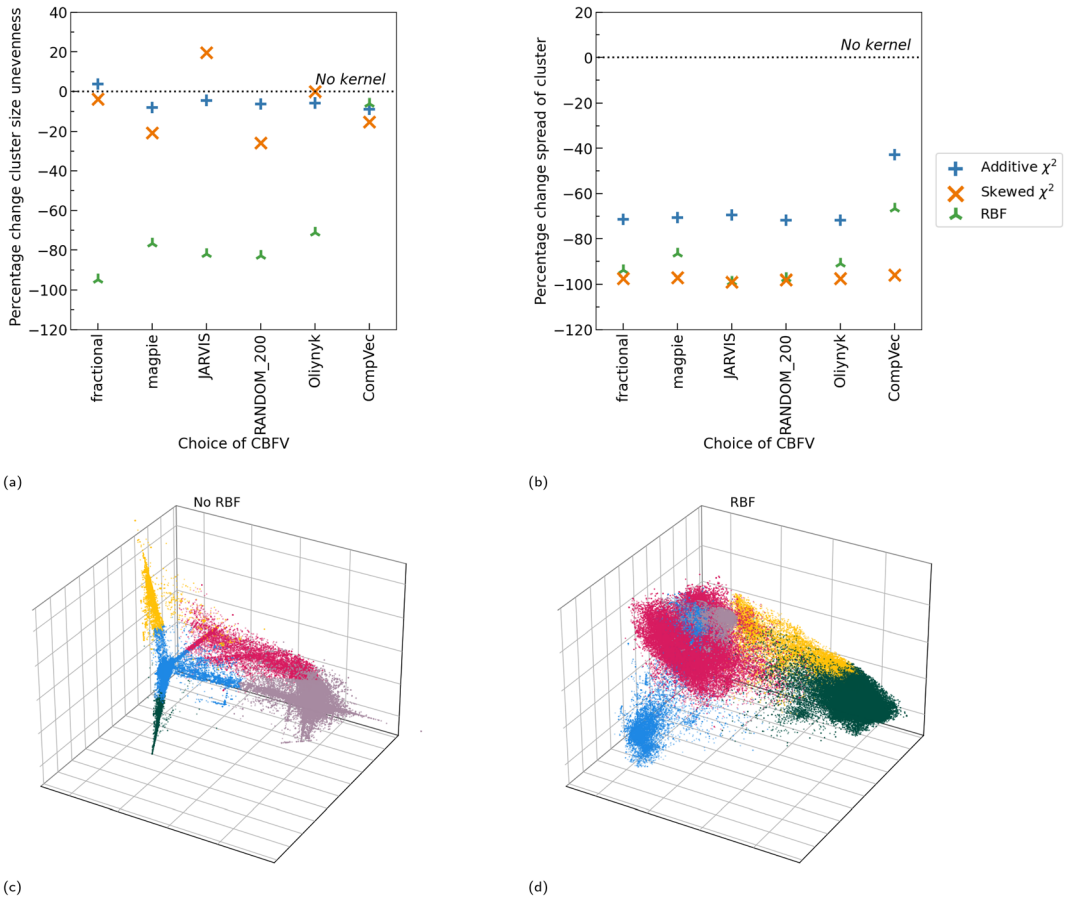


Fig. 5 Demonstration of the effect of kernel methods on clustering of compositions in the ICSD. (a) Changes in standard deviation of cluster size found by K-means clustering of ICSD ($k = 5$) with application of kernel methods. Most of the time, application of kernel methods reduces the variation between cluster sizes. This effect is most pronounced with the basis function (RBF) kernel. (b) Variation in cluster spread for K-means clustering of ICSD ($k = 5$). Application of kernel methods reduces the spread in Euclidean space within a cluster. This effect is most pronounced with skewed χ^2 and RBF. (c) To visualise these results, PCA was used to generate the first three principal components of all compositions in the ICSD featurised using a CompVec. Colours correspond to clusters found by K-means ($k = 5$) clustering on this representation. Inspection of these clusters reveals highly anisotropic clusters with no meaningful boundaries in the data to unambiguously separate clusters. (d) The first three principal components found when examining an RBF translation of the ICSD (featurised using CompVec), points are coloured according to clusters found by K-means ($k = 5$) applied to the kernelised data. The application of an RBF (as defined in Section 2.3) to every composition vector in the ICSD (before clustering) leads to clusters that are more isotropic with more clearly resolved boundaries between clusters.

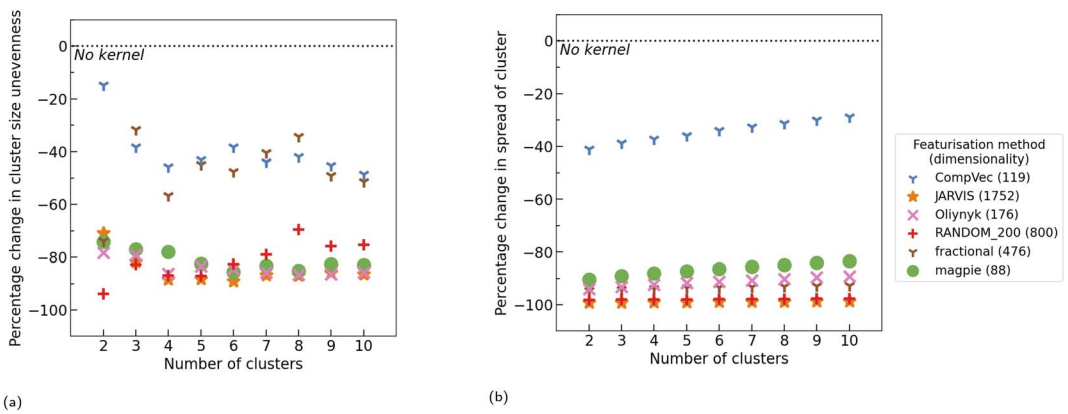


Fig. 6 Effect of radial basis function (RBF) on standard deviation of cluster sizes (cluster size unevenness) and spread of cluster sizes. This is performed using K-means clustering with different values of k. (a) RBF leads to more evenly sized clusters for all featurisation methods and nearly all values of k (b) RBF leads to more compact clusters (*i.e.*, smaller average Euclidean distance between points within a cluster) for all featurisation methods and all values of k.

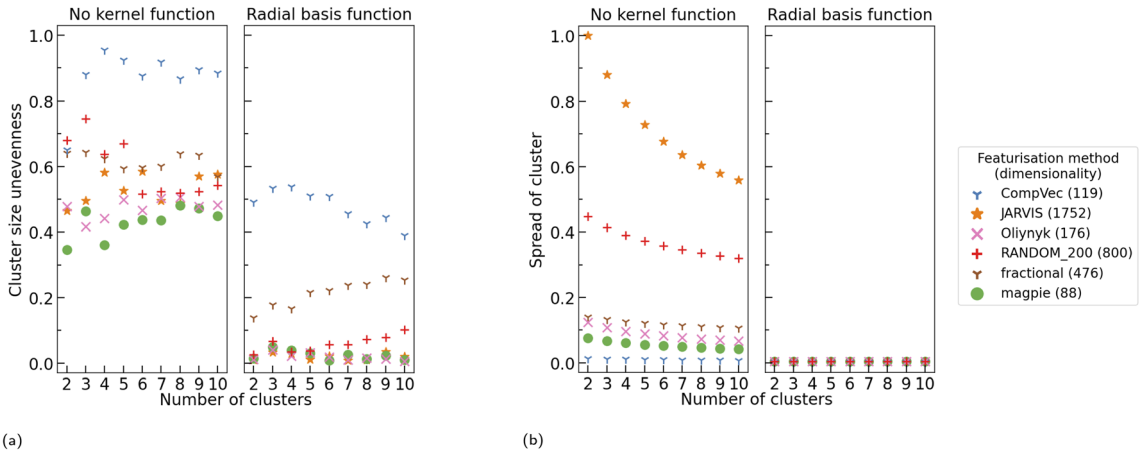


Fig. 7 Mean cluster size unevenness and spread of clusters found by *K*-means when clustering different representations of datasets. Measurements are normalised to between one and zero on a per dataset basis, as different datasets would be expected to cluster with different amounts of ease. The normalised values are then averaged across different datasets for each representation and value of *k*. (a) Clusters are generally more even in domain knowledge based representations as measured by the standard deviation in cluster sizes. (b) Without application of kernel function, spread of clusters as measured by the average distance between a point in a cluster and its centroid correlates to the size of the representation with the exception of CompVec which has the tightest clusters. Application of radial basis function makes this trend insignificant.

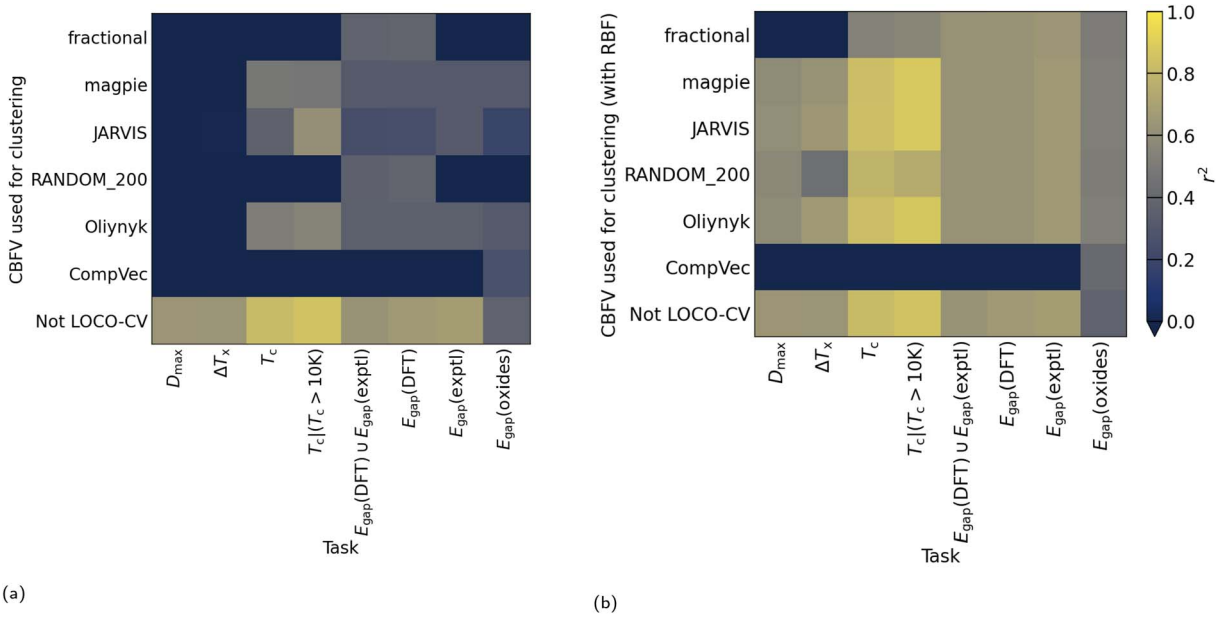


Fig. 8 Performance of random forests in regression tasks to compare evaluation regimens, measured using r^2. These random forests are evaluated with LOCO-CV (labelled with the CBFV used for *K*-means clustering), as well as a traditional 80 : 20 train:test split (labelled "Not LOCO-CV"). Importantly, in LOCO-CV, the representation used for *K*-means clustering is independent of that used for training. Accordingly, all models are trained using CompVec CBFV to remove training representation as a confounding variable. (a) Without the application of RBFs, the same random forest model which performs well in traditional 80 : 20 split training regimen often fails to learn trends in the data when evaluating with LOCO-CV, leading to low values of r^2. (b) Application of RBF to CBFVs before *K*-means clustering for LOCO-CV results in fewer models failing to learn trends in the data, leading to higher values of r^2.

experiments were carried out to measure the cluster size variance of random projections of compositions both with and without application of kernel methods.

Without application of kernel functions, when each CBFV was compared to a random projection of equal size (Fig. 9a), using random projections of composition vectors did, more often than not, result in more evenly sized clusters than CompVec, but less evenly sized clusters than all other CBFVs investigated. However, no representation (either random projection or CBFV) universally resulted in more even clusters. Comparing the best performing size of random projections (88 dimensions) with other CBFVs without any kernel methods did

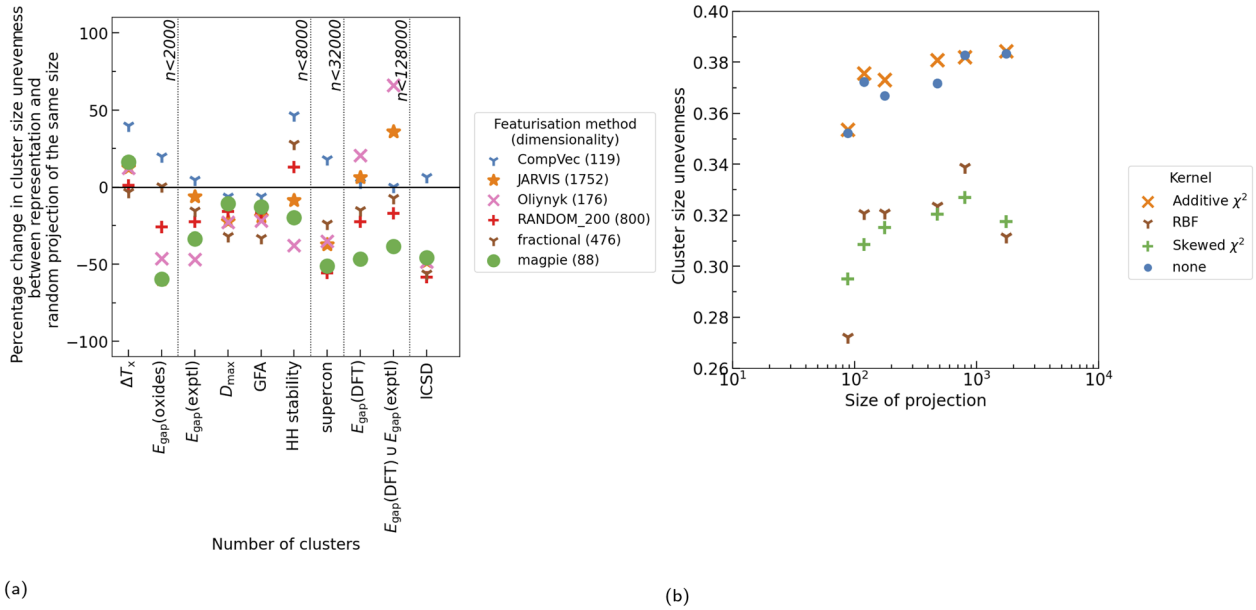


Fig. 9 (a) Reduction in cluster size unevenness (standard deviation in cluster size) of different CBFVs when compared to equal sized random projections of composition vectors across different datasets with no kernel applied. While random projection consistently outperformed CompVec, all other CBFVs form more even clusters than an equally sized random projection. (b) Average cluster size unevenness found using *K*-means clustering on datasets featurised using random projections of various sizes. Cluster size variances are normalised between 1 and 0 for each dataset (as different datasets would be expected to cluster with different amounts of ease), and then averaged for each size of random projection and each kernel. RBF and skewed χ^2 is seen to reduce cluster size unevenness, with the projections of approximately 100 dimensions performing better than larger projections.

narrow the differences in cluster size unevenness (Fig. S3b†), however other CBFVs still outperformed random projections in several datasets.

Radial basis, additive χ^2, and skewed χ^2 functions were applied to these projections before clustering using *K*-means. The resulting clusters were compared to those found without any kernel methods, showing that RBF and skewed χ^2 did reduce cluster size unevenness (Fig. 9b). However, these results still do not create a consistent pattern of either outperforming or underperforming the cluster size unevenness found by applying RBF to CBFVs (Fig. S3a†). As no representation universally results in more even clusters, a variety of CBFVs and random projections should be investigated when choosing the best representation for clustering a dataset. Application of kernel methods such as RBF are advantageous in this context regardless of representation.

4 Discussion

Recreation of studies discussed in Section 3.1 shows that, broadly speaking, featurisation methods used in research are not necessarily advantageous over random projections, especially for tasks that are not related to band gaps. Machine learning led research in materials science often aims to highlight the success of a machine learning model either in a materials discovery pipeline, as a proof of concept that a model can learn from a given dataset, or a proof of concept that a property can be predicted. As such, the exact implementation of a CBFV and its effectiveness when compared to other CBFVs are often not included in the main text of a paper. Comparison studies thus facilitate evaluation of the impact of the CBFVs on ML performance.

With modern libraries such as matminer,[10] creating new featurisation methods and changing existing ones is straightforward. The engineered featurisation methods show no advantage over more widely used, or simpler alternatives, in the tasks considered here.

Both findings here and in previous work suggest that for sufficiently large and balanced datasets, domain knowledge in CBFVs yields only small advantage.[5] Promising results in representation learning could further reduce these advantages,[45] which means the question as to whether these small advantages of feature engineered CBFVs justify the difficulty in comparison between the models using them is an open one.

Choice of representation for a supervised ML algorithm may be influenced by the extent to which the goal of the algorithm is to maximise predictive accuracy for a property (*e.g.*, to screen potential candidates for synthesis), and the extent to which the goal is to gain insight into the causes of that property. Linked to this consideration is the question of whether domain knowledge features are being used as proxy for the composition, or whether the composition is a proxy for the properties of a material which are quantified by the domain knowledge features.

For example, a model trained to predict whether a superconductor has a T_c greater than 30 K could be trained on a CBFV and find that the number of d electrons is an important indicator for this property. A similar model could be trained using

a CompVec representation and find that containing Cu is an important indicator for this property. Whether the number of d electrons is serving as proxy for the presence of Cu in a material or the presence of Cu in a material is a serving as proxy for the number of d electrons is a matter of perspective. Bearing this difference in perspective in mind may help guide towards use of a representation which is best suited for the workflow in which a machine learning algorithm is being used. If we use ML to gain insight into the causes of properties and phenomena, then examining the importance of different domain knowledge areas in a CBFV for an algorithm will allow us to do that. This would suggest that the task becomes a matter of finding the best set of properties for an element to adequately explain how it interacts with the chemistries of a compound. At this point experimenting with various combinations of elemental properties becomes appealing. However, to justify this approach adequate analysis of which properties are important is needed.

When choosing a representation to maximise predictive accuracy, domain knowledge seems to provide some advantage for some tasks examined here (particularly band gap prediction tasks). However we do not think this evidence, nor that found in previous work,[5] is sufficient to reject featurisation methods without domain knowledge such as fractional encoding of composition or random projections, for more complex or parameter dependant algorithms. When using a CBFV, random projection offers a helpful baseline for performance as it is simple to implement and works fairly well. Their single hyperparameter is the size of the projection, which allows one to draw conclusions as to the usefulness of a CBFV under investigation without introducing the size of a representation as a contributing factor for its performance.

Extrapolatory power is particularly pertinent in the materials discovery field, thus previous work presented LOCO-CV as a way to estimate the extrapolatory power of a supervised machine learning algorithm.[8] LOCO-CV (along with many other linear algorithms such as principal component analysis), relies on linear separability in the data. We show that, regardless of representation being used, kernels such as RBF are advantageous in reducing cluster size unevenness, and so should be strongly considered where such linear algorithms are applied. This reduction in cluster size unevenness tackles previously discussed caveats to LOCO-CV and results in more reliable model training (Fig. 8).

We examine the use of random projections to featurise chemical compositions to be used with kernelised LOCO-CV. As for other CBFVs examined, random projections used in conjunction with kernel methods produce more even clusters than without kernel methods. However, no representation (either CBFV or random projection) consistently resulted in more even clusters than all other representations. While most of the time CBFVs found more even clusters than random projections (with the exception of CompVec), these findings were not universal across datasets tested. Kernel methods applied to random projections resulted in cluster sizes being even enough so as to be useable in the LOCO-CV algorithm without negatively impacting conclusions drawn from measurements taken using this method.

Random projections and kernelised LOCO-CV can be used together to create a generalised workflow for evaluating the extrapolatory power of a supervised machine learning algorithm, which can be used regardless of input representation to the machine learning algorithm in question. This can be combined with using a random projection as input representation to the machine learning algorithm to see a baseline measure of extrapolatory power which prospective CBFVs can be compared against to measure their usefulness.

5 Conclusion

We demonstrate random projections are a generic and powerful way to featurise compositions for material property prediction. This is motivated by fundamental principles discussed in the Johnson–Lendenstrauss lemma;[29] randomly projecting a composition vector can be used to move such vectors into a different dimensional space while preserving relationships between points in a dataset (within some error). These random projections have only a single hyperparameter (the size of the projection), which allows us to isolate the relationships between the dimensionality of a representation, and the predictive performance of algorithms trained using that representation. Random projections can be used as a baseline representation to examine what benefit is added by domain knowledge imbued into CBFVs.

We investigate how common CBFVs could be used in ten property prediction tasks from literature, in order to establish what advantage domain knowledge offers in constructing such vectors. With the notable exception of band gap prediction tasks, CBFVs engineered to incorporate domain knowledge do not substantially outperform an equal sized random projection for most prediction tasks investigated here. If the purpose of an ML model is to maximise predictive performance, the choice of using one of many complex representations (*e.g.*, CBFVs) should be justified by demonstrating an advantage over a random projection of the same size.

We present kernelised LOCO-CV to overcome issues with imbalanced cluster sizes that often occur when performing linear clustering on material sciences datasets. The application of kernel methods, such as the RBF examined here, to data before *K*-means clustering leads to more even cluster sizes across many different datasets and input representations. Further, using these kernel-modified clusters in LOCO-CV led to more reliable model training in the models examined here. Applying kernels in LOCO-CV is independent of representations used by a supervised machine learning algorithm, so we strongly suggest that researchers looking to deploy LOCO-CV use the kernelised version presented here. Both random projections and kernelised LOCO-CV can be implemented independently or together.

We trained over 70 random forest models across ten property predictions tasks found in the materials science literature to show that random projections are a reliable baseline to use when evaluating a CBFV. We have also evaluated over 36 000 *K*-

means clustering applications, on the datasets used in these tasks as well as on the ICSD, and have shown that applying kernel functions to these data before *K*-means clustering results in more evenly sized clusters, and more reliable model training when these clusters are used in LOCO-CV. Our findings provide a basis for materials scientists in selecting and evaluating representations and laying out evaluation workflows.

6 Methods

Above experiments were implemented in Python using RF, *K*-means clustering and kernel method algorithms from the sci-kit learn library.[12] Hyperparameters of all sci-kit learn algorithms were set to default as of version 2.4.1, with the exception of the value of *k* for *K*-means clustering which was varied between 2 and 10 as needed for the LOCO-CV algorithm. While data standardisation was sometimes done before application of *K*-means clustering (as detailed in the ESI Section S1†), data standardisation was not done before application use of RFs as by their nature RFs consider dimensions independently making such standardisation redundant.

Graphs were plotted with the MatPlotLib library[46] with the exception of Fig. 8 which was also uses the Seaborn library.[47] Featurisation was done using the utilities provided with the github associated with Murdock *et al.*,[5] with the exception of CompVec which was implemented from scratch, and case study specific featurisations, which were obtained in ESI† for the relevant case study. All implementations, are made available through the associated git repository as are data used in this study.[40]

Data availability

1. Code and data (or scripts to download data) associated with Section 3 of this paper can be found at **https://github.com/lrcfmd/KernelisedLOCO-CV**.

2. Section 3.1 was carried out using data publicly available from the following sources:

(i). **https://github.com/vstanev1/Supercon**.

(ii). **https://pubs.acs.org/doi/suppl/10.1021/acs.jpcb.7b05296/suppl_file/jp7b05296_si_001.zip**.

(iii). **https://github.com/WMD-group/Solar_oxides_data**.

(iv). **https://app.globus.org/file-manager?origin_id=82f1b5c6-6e9b-11e5-ba47-22000b92c6ec&origin_path=/published/publication_1106/**.

(v). **https://link.springer.com/article/10.1007/s40192-020-00178-0#Sec12** (as ESI†).

Author contributions

S. D. performed initial experiments of the effect of representation on RF performance and kernel method application on *K*-means clustering and LOCO-CV. These experiments were built upon by S. D. in discussion with V. G., D. B., M. W. G. S. D. wrote the first draft with M. W. G. and V. G. All authors contributed to the final manuscript.

Conflicts of interest

The authors have no competing interests to declare.

Acknowledgements

This research was funded by the Leverhulme Trust *via* the Leverhulme Research Centre for Functional Material Design. We thank the Leverhulme Trust for funding this work *via* the Leverhulme Research Centre for Functional Material Design. We thank Taylor Sparks, Valentin Stanev, and Aron Walsh for correspondence which assisted in the reproduction of previous work.

Notes and references

1 J. Schmidt, M. R. Marques, S. Botti and M. A. Marques, *npj Comput. Mater.*, 2019, **5**, 1–36.
2 L. Ward, M. Aykol, B. Blaiszik, I. Foster, B. Meredig, J. Saal and S. Suram, *MRS Bull.*, 2018, **43**, 683–689.
3 K. T. Butler, D. W. Davies, H. Cartwright, O. Isayev and A. Walsh, *Nature*, 2018, **559**, 547–555.
4 A. Y.-T. Wang, R. J. Murdock, S. K. Kauwe, A. O. Oliynyk, A. Gurlo, J. Brgoch, K. A. Persson and T. D. Sparks, *Chem. Mater.*, 2020, **32**, 4954–4965.
5 R. J. Murdock, S. K. Kauwe, A. Y. T. Wang and T. D. Sparks, *Integr. Mater. Manuf. Innov.*, 2020, **9**, 221–227.
6 I. Wallach and A. Heifets, *J. Chem. Inf. Model.*, 2018, **58**, 916–932.
7 C. Rauer and T. Bereau, *J. Chem. Phys.*, 2020, **153**, 014101.
8 B. Meredig, E. Antono, C. Church, M. Hutchinson, J. Ling, S. Paradiso, B. Blaiszik, I. Foster, B. Gibbons, J. Hattrick-Simpers, A. Mehta and L. Ward, *Mol. Syst. Des. Eng.*, 2018, **3**, 819–825.
9 P. A. E. Murgatroyd, K. Routledge, S. Durdy, M. W. Gaultois, T. W. Surta, M. S. Dyer, J. B. Claridge, S. N. Savvin, D. Pelloquin, S. Hébert and J. Alaria, *Adv. Funct. Mater.*, 2021, 2100108.
10 L. Ward, A. Dunn, A. Faghaninia, N. Zimmermann, S. Bajaj, Q. Wang, J. Montoya, J. Chen, K. Bystrom, M. Dylla, K. Chard, M. Asta, K. Persson, G. Snyder, I. Foster and A. Jain, *Comput. Mater. Sci.*, 2018, **152**, 60–69.
11 L. Breiman, *Mach. Learn.*, 2001, **45**, 5–32.
12 F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot and E. Duchesnay, *J. Mach. Learn. Res.*, 2011, **12**, 2825–2830.
13 Y. Bengio, A. Courville and P. Vincent, *IEEE Trans. Pattern Anal. Mach. Intell.*, 2013, **35**, 1798–1828.
14 D. Jha, L. Ward, A. Paul, W. keng Liao, A. Choudhary, C. Wolverton and A. Agrawal, *Sci. Rep.*, 2018, **8**, 1–13.
15 L. Ward, A. Agrawal, A. Choudhary and C. Wolverton, *npj Comput. Mater.*, 2016, **2**, 1–7.
16 A. O. Oliynyk, E. Antono, T. D. Sparks, L. Ghadbeigi, M. W. Gaultois, B. Meredig and A. Mar, *Chem. Mater.*, 2016, **28**, 7324–7331.

17 K. Choudhary, B. DeCost and F. Tavazza, *Phys. Rev. Mater.*, 2018, **2**, 083801.
18 V. Stanev, C. Oses, A. G. Kusne, E. Rodriguez, J. Paglione, S. Curtarolo and I. Takeuchi, *npj Comput. Mater.*, 2018, **4**, 1–14.
19 F. Legrain, J. Carrete, A. Van Roekeghem, G. K. Madsen and N. Mingo, *J. Phys. Chem. B*, 2018, **122**, 625–632.
20 L. Ward, S. C. O'Keeffe, J. Stevick, G. R. Jelbert, M. Aykol and C. Wolverton, *Acta Mater.*, 2018, **159**, 102–111.
21 D. W. Davies, K. T. Butler and A. Walsh, *Chem. Mater.*, 2019, **31**, 7221–7230.
22 S. K. Kauwe, T. Welker and T. D. Sparks, *Integr. Mater. Manuf. Innov.*, 2020, **9**, 213–220.
23 R. Bellman, *Science*, 1966, **153**, 34–37.
24 S. Nembrini, I. König and M. N. Wright, *Bioinformatics*, 2018, **34**, 3711–3718.
25 A. Altmann, L. Toloşi, O. Sander and T. Lengauer, *Bioinformatics*, 2010, **26**, 1340–1347.
26 *SciKit Learn Feature selection*, **https://scikit-learn.org/stable/modules/feature_selection.html**, accessed: 2022-03-07.
27 H. Ritter and T. Kohonen, *Biol. Cybern.*, 1989, **61**(4), 241–254.
28 S. Kaski, *IEEE Int. Conf. Neural Networks*, 1998, **1**, 413–418.
29 S. Dasgupta and A. Gupta, *Random Struct. Algorithm*, 2003, **22**, 60–65.
30 S. P. Lloyd, *IEEE Trans. Inf. Theory*, 1982, **28**, 129–137.
31 D. Steinley and M. J. Brusco, *J. Classif.*, 2007, **24**, 99–121.
32 D. Pelleg and A. Moore, *IICML '00: Proceedings of the Seventeenth International Conference on Machine Learning*, 2000, vol. 1, pp. 727–734.
33 G. Hamerly and C. Elkan, *Advances in Neural Information Processing Systems*, 2003, vol. 16, pp. 281–288.
34 P. J. Rousseeuw, *J. Comput. Appl. Math.*, 1987, **20**, 53–65.
35 L. van der Maaten and G. Hinton, *J. Mach. Learn. Res.*, 2008, **9**, 2579–2605.
36 O. Maimon and L. Rokach, in *Data Mining and Knowledge Discovery Handbook*, Springer US, 2005, pp. 321–352.
37 M. Ester, H.-P. Kriegel, J. Sander and X. Xu, *Proceedings of the Second International Conference on Knowledge Discovery and Data Mining*, 1996, pp. 226–231.
38 J. Vert, K. Tsuda and B. Schölkopf, in *Kernel Methods in Computational Biology*, MIT Press Direct, 2004, ch. 2, pp. 35–70.
39 F. Li, C. Ionescu and C. Sminchisescu, *Pattern Recognition*, Berlin, Heidelberg, 2010, pp. 262–271.
40 *Github code repository*, **https://github.com/lrcfmd/KernelisedLOCO-CV**, accessed: 2022-03-07.
41 A. Jain, S. P. Ong, G. Hautier, W. Chen, W. D. Richards, S. Dacek, S. Cholia, D. Gunter, D. Skinner, G. Ceder and K. A. Persson, *APL Mater.*, 2013, **1**, 011002.
42 S. Curtarolo, W. Setyawan, G. L. Hart, M. Jahnatek, R. V. Chepulskii, R. H. Taylor, S. Wang, J. Xue, K. Yang, O. Levy, M. J. Mehl, H. T. Stokes, D. O. Demchenko and D. Morgan, *Comput. Mater. Sci.*, 2012, **58**, 218–226.
43 Y. Zhuo, A. Mansouri Tehrani and J. Brgoch, *J. Phys. Chem. Lett.*, 2018, **9**, 1668–1673.
44 C. L. Clement, S. K. Kauwe and T. D. Sparks, *Integr. Mater. Manuf. Innov.*, 2020, **9**, 153–156.
45 R. E. Goodall and A. A. Lee, *Nat. Commun.*, 2020, **11**, 6280.
46 J. D. Hunter, *Comput. Sci. Eng.*, 2007, **9**, 90–95.
47 M. L. Waskom, *J. Open Source Softw.*, 2021, **6**, 3021.