Extrapolation is not the same as interpolation

Citation for the original published paper (version of record):
Wang, Y., King, R. (2024). Extrapolation is not the same as interpolation. Machine Learning, 113(10): 8205-8232. http://dx.doi.org/10.1007/s10994-024-06591-2

(article starts on next page)

Machine Learning (2024) 113:8205–8232
https://doi.org/10.1007/s10994-024-06591-2

Extrapolation is not the same as interpolation

Yuxuan Wang[1] · **Ross D. King**[1,2,3]

Received: 2 March 2024 / Revised: 10 May 2024 / Accepted: 23 June 2024 /
Published online: 23 July 2024

Abstract
We propose a new machine learning formulation designed specifically for extrapolation. The textbook way to apply machine learning to drug design is to learn a univariate function that when a drug (structure) is input, the function outputs a real number (the activity): f(drug) → activity. However, experience in real-world drug design suggests that this formulation of the drug design problem is not quite correct. Specifically, what one is really interested in is extrapolation: predicting the activity of new drugs with higher activity than any existing ones. Our new formulation for extrapolation is based on learning a bivariate function that predicts the difference in activities of two drugs F(drug1, drug2) → difference in activity, followed by the use of ranking algorithms. This formulation is general and agnostic, suitable for finding samples with target values beyond the target value range of the training set. We applied the formulation to work with support vector machines , random forests , and Gradient Boosting Machines . We compared the formulation with standard regression on thousands of drug design datasets, gene expression datasets and material property datasets. The test set extrapolation metric was the identification of examples with greater values than the training set, and top-performing examples (within the top 10% of the whole dataset). On this metric our pairwise formulation vastly outperformed standard regression. Its proposed variations also showed a consistent outperformance. Its application in the stock selection problem further confirmed the advantage of this pairwise formulation.

Keywords Machine learning · Ranking · Extrapolation · Drug discovery

Editors: Ana Carolina Lorena, Albert Bifet, Rita P. Ribeiro.

✉ Yuxuan Wang
yw453@cam.ac.uk

Ross D. King
rk663@cam.ac.uk

1 Department of Chemical Engineering and Biotechnology, University of Cambridge, Philippa Fawcett Drive, Cambridge CB3 0AS, UK

2 Department of Computer Science and Engineering, Chalmers University, Gothenburg 412 96, Sweden

3 Alan Turing Institute, Euston Rd, London NW1 2DB, UK

1 Introduction

The original motivation for this work came from applying machine learning (ML) to drug design, specifically, quantitative structure activity relationship (QSAR) learning. The standard way to cast QSAR learning as ML is to learn a univariate function that when a drug (structure) is input, the function outputs a real number (activity): f(drug) → activity. The PubMed server lists around 20,000 papers doing this.

Experience in real-world drug discovery suggests that this formulation does not meet the real need in practice. Specifically, what one is really interested in is predicting the activity of new drugs with higher activity than any existing ones - extrapolation. N.B. extrapolation in QSAR learning has two related meanings: type one is the ability to make predictions for molecules with descriptor values (**x**) outside the applicability domain defined by the training set of the model (Fig 1a) (Kauwe et al., 2020; Tong et al., 2005; Nicolotti, 2018); type two is the identification of the "extrapolating molecules" with activities (y) beyond the range of activity values in the training data (Fig 1b) (Kauwe et al., 2020; Korff & Sander, 2022). In drug discovery both types of extrapolation are important. Extrapolating beyond the training set descriptor values enables new molecular types (maybe unpatented) to be proposed. Extrapolating beyond the highest observed y values is strongly desired to select more effective drugs. Here we focus on type two extrapolation.

Although many QSAR learning studies have reported advantageous ML methods based on their model prediction accuracy using metrics such as mean squared error, in practice the ability to produce accurate predictions is less valuable than the extrapolation ability in this type of application (Korff & Sander, 2022; Cramer, 2012). Indeed, some ML methods cannot extrapolate beyond the training sets. For example, random forest (RF) is incapable of predicting target values (y) outside the range of the training set because it gives ensembled prediction by averaging over its leaf predictions (Korff & Sander, 2022; Xiong et al., 2020). Our study is therefore motivated by the purpose of

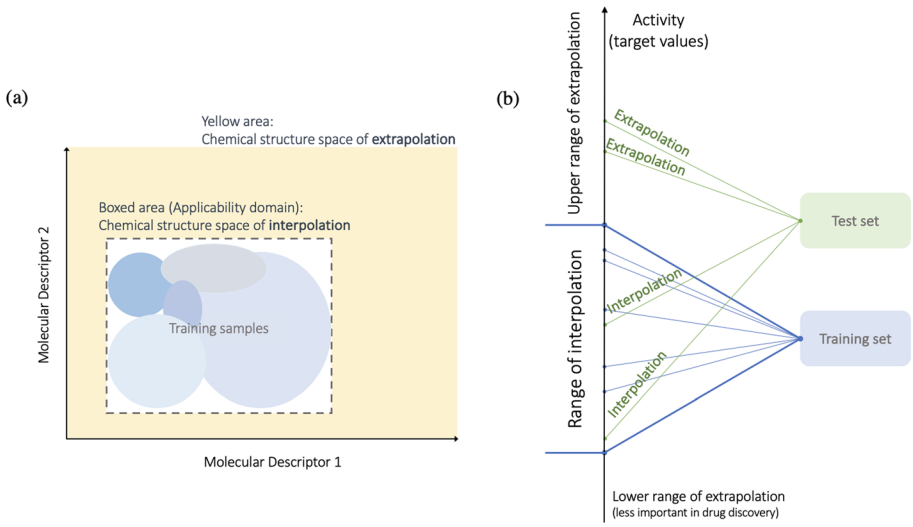


Fig. 1 The illustration of the two types of extrapolation in drug discovery. **a** type one is extrapolation outside the applicability domain, **b** type two is extrapolation outside the range of drug activities

improving ML methods to be better at finding extrapolating samples (Fig 1b). This will also be a tool that benefits many other applications, such as material sciences, dynamics modelling and system management.

Our extrapolation problem can be defined as follows. Consider a training set of N_{train} samples, its feature vectors of length N_f is $\mathbf{x} \in \mathbf{R}^{(N_f \times N_{\text{train}})}$, and its target activity values are $y \in \mathbf{R}^{N_{\text{train}}}$. Therefore, the range of the target values for the training set is between $y_{\text{train,min}}$ and $y_{\text{train,max}}$. A ML model f is then obtained so that $f(\mathbf{x}) \approx y$. Suppose there exists a test set x_{test} of size N_{test} covering a range of target values, some may be interpolating (i.e. $y_{\text{train,min}} < y_{\text{test}} < y_{\text{train,max}}$), and some may be extrapolating (i.e. $y_{\text{test}} < y_{\text{train,min}}$ or $y_{\text{test}} > y_{\text{train,max}}$). The latter ones are recognised as "extrapolating samples" and the number of them $N_{\text{extrap,true}}$ should be less than the total number of test samples N_{test}. In our study, due to the extensive application on QSAR datasets, we will restrict the reference of "extrapolating samples" to those with $y_{\text{test}} > y_{\text{train,max}}$. Therefore, the extrapolation problem will be whether the test samples with $f(\mathbf{x}_{\text{test}}) > y_{\text{train,max}}$ are truly extrapolating, or whether the model f can rank extrapolating test samples above $y_{\text{train,max}}$ if f is a ranking method. In addition, we also defined test samples as "top-performing" if their ranks are within the top 10% of the whole dataset. We would like to know if the model can rank those test samples as top 10% of the dataset of $(N_{\text{train}} + N_{\text{test}})$ samples, once the model predicts $\hat{y}_{\text{test}} = f(\mathbf{x}_{\text{test}})$ and rank the training and test samples by y_{train} and $\hat{y}_{\text{test}}$ together. Figure 2 shows an example of how the extrapolating and top-performing test samples are identified.

2 Related work

Many studies have already underscored the significance of ranking performance in drug screening. Some proposed optimising the ML method directly to enhance ranking coefficients (Agarwal et al., 2010; Rathke et al., 2011), while others suggested boosting the ranking performance from non-ML perspectives (Al-Dabbagh et al., 2017; Liu & Ning, 2017). Agarwal et al. (2010) introduced RankSVM to minimise ranking loss to maximise correctly ordered pairs of molecules. Rathke et al. (2011) developed StructRank to solve

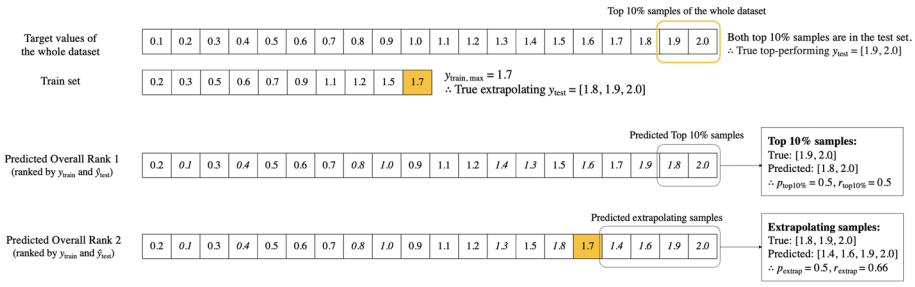


Fig. 2 An example of how extrapolating examples and top-performing samples are identified in a standard train-test split. For a given dataset, first all the top-performing samples - in this case, the top 10% - are noted. The random 50/50 train-test split is then performed. Depending on the highest target value in the train set, $y_{\text{train,max}}$, the true extrapolating test and the true top-performing test samples can be identified. After obtaining the model predictions, the training and test samples are ranked back together by y_{train} and $\hat{y}_{\text{test}}$. The italic values represent the test samples by their true y_{test} but positioned by their predicted target values, $\hat{y}_{\text{test}}$. The predicted top 10% samples are identified from this ranking, and the predicted extrapolating samples are identified as the sample ranked above $y_{\text{train,max}}$. The precision and recall metrics for each type of test samples are calculated respectively

the ranking problem by focusing on the top-k-ranked molecules. Al-Dabbagh et al. utilized quantum interference analogy for their probability ranking approach (Al-Dabbagh et al., 2017). Liu and Ning (2017) improved the ranking performance of SVMrank by leveraging assistance bioassays and compounds.

Zhang et al. successfully applied "Learning-to-rank" (LTR) from information retrieval to integrate heterogeneous data, identifying compounds by prioritising their relevance to different drug targets in a cross-target manner (Zhang et al., 2015). Although our new approach and the mentioned methods also emphasise ranking, they differ from LTR ranking algorithms. LTR models are trained to rank a fixed set of instances, focusing on optimising the relative positions of the same set of test samples for each query and extrapolation on unseen data is not required. In contrast, our approach adapts standard ML methods to distinguish sample differences explicitly and achieve extrapolation over the training set.

Recent work has emphasised the importance of extrapolation and proposed special evaluation procedures for the extrapolation performance. Kauwe et al. (2020) tested the extrapolation ability of common ML methods using properties calculated from density functional theory by keeping the top 1% of the instances in the test sets. Korff and Sander (2022) proposed to use sorted and shuffled datasets to evaluate extrapolation and interpolation performance. Xiong et al., Meredig et al. and Watson et al. have each proposed a new model validation technique to evaluate models' extrapolation performance (Xiong et al., 2020; Meredig et al., 2018; Watson et al., 2019). However, due to a lack of systematic reviews, it is unclear whether these methods are statistically meaningful. Therefore, we applied the standard k-fold cross-validation (Tong et al., 2005; Xiong et al., 2020; King et al., 2021).

This study proposes the "pairwise approach" (PA), a ML configuration approach aimed at enhancing the extrapolation ability of traditional regression methods. PA employs a pairwise model to predict differences in target values based on differences in feature vectors. In the QSAR learning context, that is to predict the differences in drugs' activity values from the differences in drugs' structural data. Extrapolation can be better achieved from the pairwise predictions.

In drug discovery, matched molecular pair analysis has been used widely to analyse the substructures that induces the key transformation in molecular properties (Tyrchan & Evertsson, 2017). As the popularity and novelty of neural networks rise, many researchers have deployed them in pairwise analyses. "Siamese" Neural Networks (SNNs), initially designed for signature verification, have found success in predicting pairwise differences, including predicting pairwise differences in drug discovery context (Fernández-Llaneza et al., 2021). This type of network has two inputs fed and processed separately before the pair of information is aggregated in the training layers (Fralish et al., 2023) (Fig. 7). Wetzel and co-workers (Wetzel et al., 2022) represented an SNN-based architecture to predict the differences in target values for pairs of samples, demonstrating its ability to compete or yield more accurate predictions for various datasets against other ML methods. Fralish et al. (2023) applied a deep SNN to predict the differences in molecular properties of pairs of drugs for ADMET drug prioritisation.

In lead optimisation, Jiménez-Luna et al. (2019) built Siamese convolutional networks on binding free energy to rank congeneric series. McNutt and Koes (2022) similarly applied Siamese convolutional networks to predict relative binding free energy ($\Delta\Delta G$) from ligand-protein binding free energies (ΔG) with increased regularisation in the latent space. (Yu et al., 2023) utilised an SNN as a part of their pairwise binding comparison network (PBCNet) for lead optimisation, which explored the conformational differences in pairs of structurally analogous small molecules against a specific target protein (McNutt & Koes, 2022). However, Tynes et al. (2021) noted the problem

of pairwise separability in SNN architectures for pairwise difference learning, especially when y_{unseen} is interpreted from the predicted pairwise differences and y_{train}. If the pairwise model is separable, i.e. the pairwise difference is predicted by mimicking the process of subtracting predictions from two individual models (e.g. two "legs" in an SNN), then it can lead to separable model loss. This problem can be represented through the training loss of the whole pairwise model being simplified as the loss of the separated models, hindering the full exploitation of the pairwise information (Tynes et al., 2021).

In Tynes et al's proposed method, namely PADRE, instead of feeding a pair of feature vectors separately, a concatenation of feature vectors together with a vector describing differences is performed before model construction, enabling the use of any ML model during training (Fig 7). Hu et al. have successfully applied PADRE using convolutional neural networks to predict the critical casting diameter of metallic glass. This work also benefits from the data augmentation brought by the pairwise expansion of the training set (Fralish et al., 2023; Wetzel et al., 2022; Tynes et al., 2021; Bao et al., 2018). These studies highlight the significance of utilizing pairs of samples and pairwise information across various domains.

3 Method

The pairwise formulation highlights the learning from the differences in feature vectors to predict the differences in the target values. As the simplest illustration, it transforms a common univariate regression model f_r in Eq. (1) to a bivariate equation in Eq. (2):

$$y \approx \hat{y} = f_r(\mathbf{x}), \tag{1}$$

$$Y \approx \hat{Y} = F_r(\mathbf{X}), \tag{2}$$

where $\mathbf{X} = g(\mathbf{x}_i, \mathbf{x}_j)$, $\mathbf{x}$ is the feature vector, i and j are the ith and jth sample in the training set, y is the target value, $\mathbf{X}$ is the transformed pairwise feature vectors, Y is the pairwise differences in target values, and g is the feature transformation function that generates the pairwise feature vectors. PADRE used this methodology (Tynes et al., 2021). The detailed notations can be found in Appendix A.

3.1 Feature transformation function

The feature transformation function g varies depending on the data type of the original feature vectors. In our previous study (Wang & King, 2023), we explored the formulation of pairwise features for Boolean inputs. For a Boolean feature, a pair of samples P_{AB} is derived from sample A (S_A) and sample B (S_B). The difference in the ith feature for this pair can be presented in one of the following ways: present in both samples ($x_{A,i} = 1, x_{B,i} = 1$), present in S_A but not in S_B ($x_{A,i} = 1, x_{B,i} = 0$), present in S_B but not in S_A ($x_{A,i} = 0, x_{B,i} = 1$), and absent from both samples ($x_{A,i} = 0, x_{B,i} = 0$). To represent each type of difference in the substructure, a unique value is assigned to the ith feature of the pair. In this way, the original number of features will be preserved. The unique values used in our experiments are (see discussion in Section A.1):

$$x_{A,i} = 1, x_{B,i} = 1 \rightarrow X_{AB,i} = 2$$
$$x_{A,i} = 1, x_{B,i} = 0 \rightarrow X_{AB,i} = 1$$
$$x_{A,i} = 0, x_{B,i} = 1 \rightarrow X_{AB,i} = -1$$
$$x_{A,i} = 0, x_{B,i} = 0 \rightarrow X_{AB,i} = 0$$

For a regressive feature input with continuous values or multiple discrete values, it is difficult to assign unique pairwise feature values for every combination as above. Therefore, its corresponding pairwise feature will be the concatenation of the difference in two feature values, $x_{A,i} - x_{B,i}$, and the first minuend feature vector $x_{A,i}$ (see discussion in Section A.2),

$$X_{AB,i} = (x_{A,i} - x_{B,i}) \oplus x_{A,i} \tag{3}$$

3.2 Extrapolation strategy

We focused on the extrapolation results and gave a lower priority for numerical accuracy. We added the following modifications to the bivariant learning problem above so that the extrapolation performance and the ranking performance near the top end can be enhanced.

3.2.1 Variation 1 (PA-V1)

We applied the ranking algorithm, Trueskill, on the predicted pairwise differences to rank the training and test samples together. Trueskill was originally developed to rank players in the game “Halo” (Herbrich et al., 2007). It can accommodate variations in performance and skill levels, allowing for handling potential conflicts in match outcomes. Each predicted difference is treated as a "game match" between two samples. If the difference between S_A and S_B is greater than 0, then S_A wins S_B. Trueskill updates a "league table of samples" to determine rankings, from which the predicted extrapolating or top-performing test samples are found.

Instead of learning a regression model f_r for the pairwise differences Y, we trained a classification model f_c to predict the signed differences in target values, sign(Y). Since Trueskill ranks samples based on wins and losses, the accuracy of predicting wins ($Y_{AB} > 0$) or losses ($Y_{AB} < 0$) is crucial compared to numerical accuracy. We have noticed that training the pairwise model via classification yields higher accuracy in predicted signs compared to extracting signs from numerical differences via regression. In other words, the accuracy of $\text{sign}(Y)^{\text{pred}}$ is higher than that of $\text{sign}(Y^{\text{pred}})$. This may be because some pairs have identical feature vector differences but distinct target value differences. Despite some loss of information when taking the signs, the training of the classification model may encounter less “noise” in pairwise target values than the regression model. For a generic ranking algorithm, correct results of win or loss are more important in deciding the rank of the samples than the more accurate numerical score differences with potentially wrong signs. Therefore, training the pairwise model via classification and a generic version of the rating algorithm was used, transforming the problem into:

$$\text{sign}(Y) \approx \text{sign}(Y)^{\text{pred}} = f_c(\mathbf{X}), \tag{4}$$

$$r \approx \hat{r} = R(\text{sign}(Y)^{\text{pred}}), \tag{5}$$

where $\mathbf{X} = g(x_i, x_j)$, r is the rank of samples with respect to their corresponding target values y and R is the ranking algorithm which in this study is the Trueskill algorithm (Herbrich et al., 2007), TrueSkill .

3.2.2 Variation 2 (PA-V2)

In this variation, we utilise the previously discarded absolute differences as part of the ranking process. Guo et al. (2012) have proposed a new version of Trueskill which takes account of the absolute match scores for ranking, named "Score-based Trueskill". Instead of feeding the ranking algorithm with only wins (sign(Y_{AB}) = 1) and losses (sign(Y_{AB}) = -1), we can feed continuous values to represent how much S_A wins over S_B. Despite some loss of accuracy in the signed differences, this algorithm allows us to utilise as much prediction information as possible to predict the ranking. Hence, we used a regression model to predict the Δy between samples. The predictions were then fed directly to the Score-based Trueskill to obtain a ranking of the datasets. The variation has the following transformation:

$$Y \approx \hat{Y} = f_r(\mathbf{X}), \tag{6}$$

$$r \approx \hat{r} = R_s(\hat{Y}), \tag{7}$$

where $\mathbf{X} = g(x_i, x_j)$, r is the rank of samples with respect to their corresponding target values y and R_s is the ranking algorithm which in this case is the Score-based Trueskill algorithm (Guo et al., 2012).

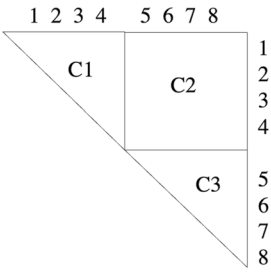
3.3 Notations for variations of the pairwise approach

Suppose a dataset is split into a training set of size N_{train} and a test set of size N_{test}. The training samples are paired via permutation, creating N^2_{train} pairwise training pairs. This type of pair is referred to as a C1-type training pair in this study. The test pairs can be obtained in two ways: (1) C3-type test pairs: generated from a permutation of test molecules, giving N^2_{test} test pairs; (2) C2-type test pairs: generated from pairing test molecules with training molecules, giving $2N_{\text{train}}N_{\text{test}}$ test pairs. The naming of the pair types follows

Fig. 3 The example of partitioning the pair space using the pair notation (Park & Marcotte, 2012)

Training data points	Test data points
1	5
2	6
3	7
4	8

Partitioning of the test pair space into 3 classes



the notation in the work by Park and Marcotte (2012) which considers the amount of shared information between training and test data within a pair (see Fig. 3).

In an extrapolation task, the relationship between the test samples and the training samples is important for comparing the training and test data to find the extrapolating samples. So, despite the existence of C3-type test pairs, using them to rank alone can only tell the relative ranks within the test set. On the other hand, C2-type test pairs describe the relative differences between training and test samples. These are better suited for the extrapolation task. Therefore, in the following experiments on extrapolation, C2-type test pairs or C2-type + C3-type test pairs will be primarily used to rank.

Due to the use of combinations of different extrapolation strategies and pairs for ranking, we will letter-code each specific arrangement. For example, PA-V1-C2 means the pairwise approach that uses the standard Trueskill with C2-type test pairs as ranking inputs. PA-V2-C2C3 means the pairwise approach that uses the Score-based Trueskill with C2-type test pairs + C3-type test pairs as ranking inputs.

3.4 Machine learning methods & evaluation metrics

Our pairwise formulation is potentially ML method agnostic. We, therefore, utilised the most common ML methods applied to QSAR learning: support vector machines (SVMs), random forests (RFs), Gradient Boosting Machine (XGBs) and *k*-nearest neighbours (KNN). We did not evaluate the pairwise approach using deep learning algorithms, this was because neural networks are not particularly well suited to traditional QSAR problems (Olier et al., 2018), and because the amount of chemoinformatic data is generally too small for deep learning methods to be effective. The ML methods used in this study are all based on the open-source ML python library, scikit-learn (Pedregosa et al., 2011). When a ML method is used to compare the standard and pairwise models, it is used with the default parameter setting from scikit-learn.

Before training any ML model, a basic feature selection is performed to reduce the feature space and accelerate the learning. For a given dataset, the features were removed if they had the same feature value assigned to every sample in a dataset. The features that repeat to have the same pattern for all the samples were also removed.

To evaluate the extrapolation ability of a ML method, metrics other than the traditional evaluation metrics, such as mean squared error and R-squared, are required. This is because these metrics are designed to cover predicted results over the whole test set, resulting in an averaged performance evaluation for both interpolation and extrapolation. In a random split in cross-validation, the test set usually contains more interpolating samples than extrapolating samples. Therefore, these metrics are good for evaluating the interpolation power of a model, but not very informative in terms of its extrapolation power (Xiong et al., 2020). In this study, we decided to adopt the classification metrics of precision, recall and f1 score to count the identification of extrapolating and top-performing samples (Kauwe et al., 2020; Xiong et al., 2020). This will give a more direct view of how useful a ML method is in an application where identifying such samples is highly desired.

The precision (p), recall (r) and f1 score ($f1$) are calculated for two different types of identification, extrapolating samples and top 10% samples. The number of extrapolating samples depends on the split of training and test sets. A test sample will be classified as "extrapolating" (subscripted as *extrap*) if its true target value is greater than the maximum target value in the training set. After ranking the whole datasets by their true target values, a test sample will be classified as "top-performing" (subscripted as *top10%*) if it is ranked

as one of the top 10%. The model-predicted extrapolating and top-performing samples will be obtained in the same way, except for the ranking positions of the test sample determined by their predicted target values, $\hat{y}_{\text{test}}$. The true list and the predicted list of each type of samples will be compared in every run to evaluate the corresponding metrics (see Fig. 2 for the example calculation). When comparing the two lists, the true positives (TP), false positives (FP), and false negatives (FN) can be identified, from which the precision, recall and f1 score are calculated:

$$\text{precision} = \frac{\text{TP}}{\text{TP} + \text{FP}}, \quad \text{recall} = \frac{\text{TP}}{\text{TP} + \text{FN}}, \quad \text{f1 score} = 2\frac{pr}{p + r}.$$

4 Results

4.1 Application of the pairwise approach on Boolean datasets

Our extrapolation experiments on 1436 Boolean ChEMBL datasets (see Appendix B.4.1) showed a clear advantage of the pairwise approach over the standard approach (Table 1(a) and Fig. 4). The ChEMBL datasets were sorted by size and experimented sequentially via 10-fold cross validation. When comparing the two approaches, the standard approach uses the regression version of a ML method to predict target values y and rank the test samples with training samples by predicted target values, while the pairwise approach uses the classification version of that ML method to predict sign(Y_{C2}) to rank the whole dataset. The pairwise approach used PA-V1-C2 variation, the very first proposal on the pairwise formulation for ranking purposes.

It was found that the pairwise approach was much better at recognising the extrapolating and top-performing molecules than the standard approach. For all the three ML methods (RF, SVM and XGB) tested, the pairwise approach almost always found equally or more

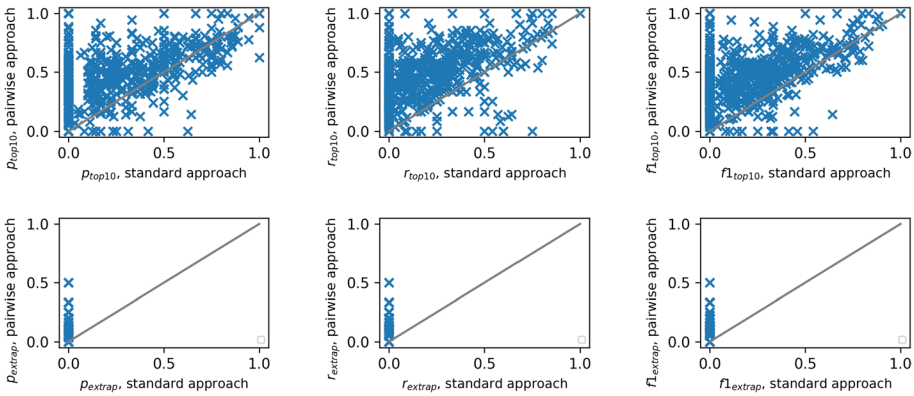


Fig. 4 The six metrics of the pairwise approach (PA-V1-C2) and of the standard approach on ChEMBL datasets using SVM. Each sub-figure represents one of the metrics evaluated over 1436 datasets. For each metric in each subfigure, the metric of the pairwise approach is plotted against that of the standard approach. The line of equal performance is plotted as the grey line. The markers above the line of equal performance indicate those datasets of outperformance by the pairwise approach

extrapolating molecules than the standard approach (Table 1(a)(i)). It can also identify more test molecules ranked within the top 10% of the dataset most of the time, as shown by a high percentage for $r_{\text{top10\%}}$. Its outperformance in $p_{\text{top10\%}}$ is not as good as that in $r_{\text{top10\%}}$, but is still overall better than the standard approach. It was also noted that this outperformance is less outstanding for XGB or larger datasets. This suggests that the ratio of false positives in the top-performing molecules using the pairwise approach can sometimes be similar to that of the standard approach. At the same time, the pairwise approach often caused a greater increase in recall, which means it proposed more true positives. Hence, despite an outperformance in $p_{\text{top10\%}}$, the pairwise approach could propose slightly more false positives together with more true positives.

As extrapolating molecules do not necessarily exist in every train-test split, many datasets were showing $p_{\text{extrap}} = r_{\text{extrap}} = f1_{\text{extrap}} = 0$ or non-existing. Therefore, to illustrate outperformance, the datasets showing equal performance were removed. The percentage of datasets suggesting the pairwise approach outperformed the standard approach was recalculated for the rest of the datasets (Table 1(ii)). Across three ML methods tested, the pairwise approach indeed outperformed the standard approach in finding both the extrapolating and top-performing molecules. The results also suggested that RF or XGB had less outperformance than SVM. Through further investigation, we found that the difference among ML methods was due to the variation in extrapolation performance by the standard approach. All three methods performed equally badly on extrapolating samples, which indicates their incapability to extrapolate beyond the range of the training target values. However, for top-performing samples the standard approach using RF or XGB can evidently produce higher extrapolation metrics than SVM. At the same time, the pairwise approach performed similarly via both ML methods. This gives rise to the higher percentage of datasets showing the pairwise approach was better with SVM in Table 1(a) and Fig. 4. Despite the SVM having been shown to be a good-performing ML method for QSAR learning, we considered the following reasons to account for its bad performance in our experiments. The poor performance might come from the relatively small dataset sizes, where the curse of high dimensionality can show up due to the length of the fingerprint, leading to potential overfitting. Despite applying feature pre-processing to reduce the number of features, the remaining feature size is still much larger than the sample size. Tree-based methods certainly might suffer from the same issue. The randomness in sub-trees could mitigate the effect slightly. After the pairwise formulation is carried out, the data size is augmented while keeping the same number of features, and the performance of SVM becomes comparable to RF and XGB. The second reason might be the use of the radial basis function kernel, which calculates the Euclidean distances between points. During standard learning especially for small datasets, the Boolean features might give rise to less informative distance similarity measurement, leading to a poor performance. The outperformance of the RF is also supported by the comprehensive comparison studies applied to the same QSAR data by Olier et al. (2018).

Apart from a statistical overview of the extrapolation power of the pairwise approach, we examined its performance versus the size of the datasets. Fig. 5 shows an example of the increase in $f1_{\text{top10\%}}$ versus dataset size for the experiments with RF. The plots for other metrics showed a similar trend, that is the pairwise approach is more advantageous on smaller datasets, indicated by more data points above the line of $\Delta f1_{\text{top10\%}} = 0$ when the dataset size is less than 200. This is mainly due to the standard approach learning better when the dataset was larger, reducing the difference between the two approaches.

We also applied the PA-V2 pairwise approach on the same set of datasets to compare with the PA-V1 version, where the difference is the use of regressive pairwise predictions

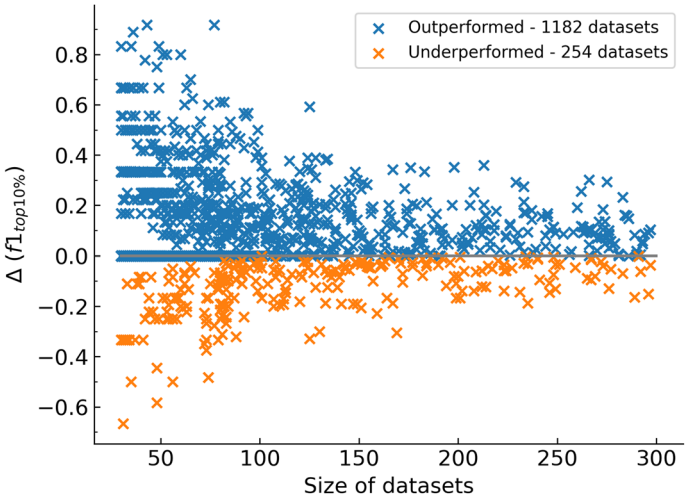


Fig. 5 The increase in f1 score for top-performing molecules by the pairwise approach (PA-V1-C2) versus the size of datasets with RF for 1436 Boolean ChEMBL datasets. On y-axis, $\Delta f1_{\text{top10\%}} = f1_{\text{top10\%}}(\text{pairwise}) - f1_{\text{top10\%}}(\text{standard})$

and the Score-based Trueskill to obtain the samples' ranking. Comparing the results in Table 1(a) and (e), the two variations are very similar in terms of their performance over the standard approach. PA-V2-C2 had slightly higher extrapolation metrics than PA-V1-C2 in terms of the number of datasets showing an outperformance of our approach. Comparing Table 1(e)(i) and (e)(ii), we can see that PA-V2-C2C3 can marginally increase the percentage of outperforming datasets thanks to the addition of extra pairwise prediction information from C3-type test pairs.

To test the generality of the paired formulation, we applied the same comparison experiment to a set of human gene expression datasets (see Appendix B.4.2). The datasets were used by Olier et al. in a transformational ML study (Olier et al., 2021). Because each dataset contains 118050 rows of experimental conditions (samples), if the pairwise approach is applied for this size, the pairwise training set will be too large to train given any reasonable computational resources. We therefore decided to randomly sample a size 100 or 200 from each of the 978 gene datasets to compare the extrapolation performance. The extrapolation metrics were evaluated for the standard and the pairwise approach across four ML methods, random forest (RF), support vector machine (SVM), *k*-nearest neighbour (KNN) and gradient boosting machine (XGB). PA-V1-C2 pairwise strategy was used in this set of experiments and 10-fold cross validation was used in the evaluations.

It can be seen in Table 1(b) that for the gene expression datasets, the pairwise approach followed a similar trend as seen in the Boolean ChEMBL experiments to outperform the standard approach. When the size of the datasets increased from 100 to 200, some of the extrapolation metrics decreased. This is also because the standard approach improved its learning through the additional data at a rate slightly greater than the pairwise approach, resulting in a decrease in the percentage of datasets showing outperformance. This is consistent with observations from Fig. 5.

We have shown that pairwise learning and Trueskill work well together to improve the extrapolation. To further testify that pairwise learning of the sample differences plays an

Table 1 The percentage of datasets showing the pairwise approach with different variation arrangements (i) had an equal or better performance than the standard approach, i.e., metric(pairwise)>=metric(standard), (ii) was better than the standard approach, i.e., metric(pairwise) > metric(standard), excluding datasets showing equal performance

Datasets	(a) Boolean ChEMBL 1436 datasets						(b) Gene expression datasets 978 datasets					
Percentage type	(i)			(ii)			(i)					
PA arrangement	V1-C2						V1-C2					
ML Methods	RF	SVM	XGB	RF	SVM	XGB	RF-100	RF-200	KNN-100	KNN-200	SVM-100	XGB-100
p_{extrap}	99.8%	100.0%	99.4%	99.2%	100.0%	96.6%	100.0%	100.0%	100.0%	100.0%	100.0%	99.9%
r_{extrap}	99.9%	100.0%	99.5%	99.6%	100.0%	97.4%	100.0%	100.0%	100.0%	100.0%	100.0%	99.9%
$f1_{extrap}$	99.9%	100.0%	99.4%	99.2%	100.0%	97.0%	100.0%	100.0%	100.0%	100.0%	100.0%	99.9%
$p_{top10\%}$	78.1%	92.4%	72.4%	66.8%	89.7%	58.8%	87.6%	71.6%	86.3%	77.2%	86.3%	76.6%
$r_{top10\%}$	88.7%	97.2%	86.4%	82.4%	96.3%	78.5%	88.9%	77.3%	85.9%	80.3%	85.7%	78.2%
$f1_{top10\%}$	82.3%	95.4%	76.7%	74.3%	93.9%	66.7%	87.2%	70.6%	85.6%	76.3%	85.3%	75.3%

Datasets	(c) Regressive ChEMBL, BasicMolProp 1526 datasets						(d) Regressive ChEMBL, AllMolProp 1362 datasets					
Percentage type	(i)						(i)					
PA arrangement	V1-C2	V1-C2C3	V1-C2	V1-C2C3	V1-C2	V1-C2C3	V1-C2	V1-C2C3	V1-C2	V1-C2C3	V1-C2	V1-C2C3
ML Methods	RF		SVM		XGB		RF		SVM		XGB	
p_{extrap}	100.0%	100.0%	100.0%	100.0%	99.8%	99.8%	100.0%	100.0%	100.0%	100.0%	99.9%	99.9%
r_{extrap}	100.0%	100.0%	100.0%	100.0%	99.9%	99.9%	100.0%	100.0%	100.0%	100.0%	99.9%	99.9%
$f1_{extrap}$	100.0%	100.0%	100.0%	100.0%	99.8%	99.8%	100.0%	100.0%	100.0%	100.0%	99.9%	99.9%
$p_{top10\%}$	85.6%	86.1%	96.0%	96.1%	68.4%	68.8%	88.9%	90.3%	95.1%	95.1%	72.4%	73.2%
$r_{top10\%}$	94.3%	94.3%	99.5%	99.6%	83.0%	83.8%	95.0%	94.1%	98.9%	99.0%	80.3%	80.8%
$f1_{top10\%}$	89.8%	89.6%	98.0%	98.1%	76.0%	76.6%	91.3%	91.4%	97.0%	96.9%	74.5%	75.0%

Table 1 (continued)

Datasets	(e) Boolean ChEMBL 1436 datasets				(f) Band gap				(g) Formation energy			
Percentage type	(i)		(ii)		(i)				(i)			
PA arrangement	V2-C2	V2-C2C3	V2-C2	V2-C2C3	V1-C2	V1-C2C3	V2-C2	V2-C2C3	V1-C2	V1-C2C3	V2-C2	V2-C2C3
ML Methods	RF				RF				RF			
p_{extrap}	100.0%	100.0%	100.0%	100.0%	100.0%	100.0%	100.0%	100.0%	100.0%	100.0%	100.0%	100.0%
r_{extrap}	100.0%	100.0%	100.0%	100.0%	100.0%	100.0%	100.0%	100.0%	100.0%	100.0%	100.0%	100.0%
$f1_{extrap}$	100.0%	100.0%	100.0%	100.0%	100.0%	100.0%	100.0%	100.0%	100.0%	100.0%	100.0%	100.0%
$p_{top10\%}$	79.7%	79.3%	71.6%	71.7%	**51.7%**	**53.3%**	**53.3%**	**56.7%**	65.0%	68.3%	65.0%	65.0%
$r_{top10\%}$	92.5%	91.8%	89.1%	88.4%	100.0%	100.0%	98.3%	95.0%	80.0%	81.7%	96.7%	86.7%
$f1_{top10\%}$	84.5%	83.5%	79.0%	78.3%	76.7%	83.3%	83.3%	86.7%	73.3%	76.7%	90.0%	81.7%

All the values have a binomial p value < 0.05 except the ones in bold

important part in the overall improvement, we additionally tested the performance with a simpler rating algorithm, Elo's rating system. It is an algorithm developed by Arpad Elo originally for chess competitions (Lehmann & Wohlrabe, 2017). The ranking of the players is according to the actual win or loss and the expected probability of win or loss, which is calculated via a logistic curve. Even with Elo's rating algorithm, we still observed a clear outperformance by the pairwise approach over the standard approach (see Appendix Table 2). Trueskill and Elo's rating differs mainly in the balance between precision and recall. Compared to Trueskill, Elo's rating showed a lower winning percentage of datasets on precision, but a higher winning percentage on recall. Therefore, the choice of an appropriate rating algorithm should also match the objective of the application. For example, when discovering potential high-activity drugs from an untested dataset, if the false positives would heavily burden the experiments, then a ranking algorithm that gives better precision metrics should be chosen.

4.2 Application of the pairwise approach on regressive datasets

We compared the pairwise approach with the standard approach on the regressive ChEMBL datasets with molecular descriptors generated by Olier et al. (2018). The molecular descriptors are extracted by Dragon Version 6, a commercially available software library that can potentially calculate up to 4885 molecular descriptors (Mauri et al., 2006). We used two sets of regressive features to represent the ChEMBL datasets: "AllMolProp" comprising a maximum of 1447 molecular descriptors using all permitted molecular descriptors for 2D molecular structures, and "BasicMolProp", a subset of AllMolProp with a size of 43.

Similar to previous experiments, for each type of feature representation the ChEMBL datasets were sorted by size and experimented in order. We used default RF with a 10-fold cross validation to validate the result per dataset. We obtained the results from 1526 ChEMBL-BasicMolProp datasets of size from 30 to 240 (Table 1(c)) and 1362 ChEMBL-AllMolProp datasets of size from 30 to 175 (Table 1(d)). We consistently observed the pairwise approach outperforming the standard approach (see Appendix, Fig. 8). Results using SVM and XGB are also available in Table 1(c, d). Again, the pairwise approach discovered equally or more extrapolating samples in almost all the datasets. The pairwise approach PA-V1-C2C3 outperformed PA-V1-C2 on slightly more datasets. A similar conclusion from these experiments indicated the feasibility of our pairwise feature transformation for regressive features.

To extend the evaluation on other discovery problems, we assessed the model performance on two datasets of material property prediction for formation energy and band gap, which were processed and used in Xiong et al.'s study (Xiong et al., 2020) (see Appendix B.4.3). Given the large dataset size, we investigated the extrapolation behaviour of the pairwise approach on smaller subsets considering its exhibited advantage in small datasets in previous experiments. For each dataset of size 10042, we randomly sampled various sizes from 50 to 500 ten times and compared the standard approach and pairwise approach using RF with the default setting and 10-fold cross validation.

The results showed consistent outperformance by the pairwise approach as before, achieving higher extrapolation metrics for extrapolating samples for all the datasets, and higher metrics for top-performing samples for most of the datasets (see Table 1(f, g)). The analysis against dataset size showed that the pairwise approach is particularly outstanding for datasets of sizes less than 200 (Fig. 9). With increasing dataset size the rapid decrease

in precision caused the smaller outperformance margin of the pairwise approach. While it maintained its strength with high recall, the drop in precision means that it also falsely identified many non-top-performing samples. Introducing additional information from C3-type test pairs into the ranking step can slightly mitigate precision drop (Table 1(f)). Again, we confirmed the problem of the false positives in identifying the top-performing samples in this set of experiments. The differences between PA-V1 and PA-V2 methods were also revealed. For top-performing samples, the PA-V2 method can notably improve all three extrapolation metrics, especially for datasets of size under 200. In contrast, for extrapolating samples, the PA-V2 method kept a similar performance for datasets of size under 200, but showed a reduction in all three metrics for larger datasets (Fig. 10).

Furthermore, we also confirmed the improvement in MSE in $\hat{y}_{\text{test}}$ using the pairwise differences and target value of the training samples, as mentioned in several studies (Wetzel et al., 2022; Tynes et al., 2021). After training a regression model on the pairwise differences directly using Equation 6 for C2-type test pairs, we can then obtain multiple estimates for $\hat{y}_{\text{test}}$, the number of which equals twice the size of the training set:

$$\hat{y}_{\text{test},j} = y_{\text{train},i} - \hat{Y}_{C2,ij}$$
$$\hat{y}_{\text{test},j} = \hat{Y}_{C2,ji} + y_{\text{train},i}$$

By averaging these estimations, we can compare the pairwise approach and the standard approach via their $\hat{y}_{\text{test}}$. Averaging these estimations using predictions from the regressive pairwise model resulted in decreased MSE across datasets (see Fig. 11).

4.3 Application of the pairwise approach on stock selection

The pairwise approach was tested against a real-world problem beyond the scientific discovery domain. We extracted the S&P 500 stock information from Macrotrends. It comprises 503 common stocks issued by 500 large-cap companies traded on American stock exchanges, including about 80% of the American equity market by capitalisation. We assessed the model's predictive performance with two tuned ML models, RF and SVM, to conduct stock picking (see Appendix Table 4 for the hyperparameters), using 14 stock ratios proposed by Huang as features (Huang, 2012) (see Appendix Table 3). The target value was the percentage increase in the stock price over a year. The ratios and the price percentage increases were extracted from Macrotrend since 2010. Because the pairwise approach is not specifically tailored to handle the unpredictable nature of stock markets, we limited our study to years of relative stability before the pandemic in 2020.

For each Year i between 2011 and 2018, we trained a tuned regression model using the stock ratios from Year $(i-1)$ to predict the price percentage increase in Year i. Then we input the trained model with the stock ratios from Year i to predict the percentage increase in stock price in Year $(i+1)$. We selected the top 10 or 50 stocks from each set of predictions, ranked by the standard approach or the pairwise approach, to evaluate the mean annual stock return. When using RF, the entire profile over the years of interest was averaged over three repeated runs. In each repeat, a different set of random states was used in the RF models to alleviate the stochastic effect of RF as all the trees in one RF model are built on randomly bootstrapped subsets of the training set. The cumulative stock return in terms of capital growth percentage from 2011 until 2019 was plotted as shown in Fig. 6.

In three out of four experiments, each involving a different combination of ML methods and the number of selected stocks, the pairwise approach showed a clear outperformance

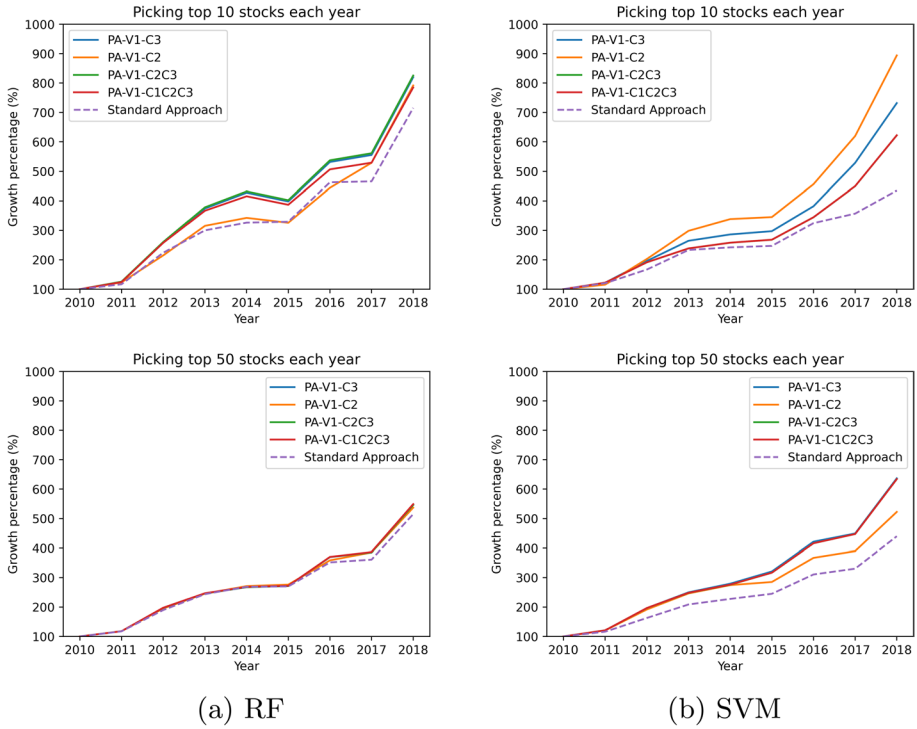


Fig. 6 The percentage growth of capital by the selection of top stocks suggested by the standard approach (dashed line) versus by the pairwise approach (solid line) with different ranking input choices using **a** tuned RF, **b** tuned SVM

over the standard approach. Notably, in the experiments with SVM models selecting the top 10 (SVM-top10), the pairwise approach can generate up to almost 460% more cumulative stock return than the standard approach by the end of the period. For RF models selecting top 10 stocks (RF-top10), the pairwise approach also achieved enhanced performance, although the improvement compared to SVM models was less pronounced due to the improvement in the prediction by the standard approach. Nevertheless, by the end of 2019, the capital increase using the pairwise approach exceeded that of the standard approach by at least 70%. Despite the better performance of the standard approach with RF, the pairwise approach still obtained an improved performance to go beyond.

In experiments selecting the top 50 stocks, overall the capital growth by the end of the investigated period was generally less than in experiments selecting the top 10. By picking more stocks, the same amount of capital was equally distributed among a larger number of stocks, resulting in less funds to spend on each stock. As a result, each year for the rising or falling stock, the earning return or capital loss was smaller, respectively. Additionally, more underperforming stocks were included in the portfolio, assuming the ratio of rising stocks to falling stocks was lower compared to when only selecting the top 10. Therefore, the average return per stock was lowered, and the total return converged towards the stock return from a random portfolio. This underscores the importance of correct extrapolation in such applications.

Although all the ranking strategies performed better than the standard approach, we have noticed that the four ranking strategies performed differently, which varied the capital return. In the SVM-top10 experiment, PA-V1-C2 achieved the highest capital return of 894%. However, in the SVM-top50 experiment, PA-V1-C2 was the weakest among the four. We believe that compared to PA-C2C3, using solely C2-type test pairs (PA-C2) gave less information about the ranking of the test samples. The relative positions of the test samples are slightly more random and uncertain, resulting in a more vigorous variation in the final performance. Therefore, for more stable and reliable performance, it is recommended to apply PA-C2C3 for any relevant applications. This conclusion is consistent with our findings in our previous experiments. Due to the small number of datasets and runs in this experiment, it was difficult to demonstrate the statistical significance in this experiment.

5 Discussion

The pairwise formulation represents a methodology that combines model reconfiguration, feature preprocessing, and post-learning refinement techniques rather than introducing a new ML algorithm. It can be fitted with multiple types of ML methods. The new formulation shifts the ML learning objective to the relationship between training and test samples. As a crucial step, the utilisation of the ranking algorithm significantly advances the fulfilment of the extrapolation purpose. Our previous work already showed that the sole use of Trueskill on the standard approach can enhance the extrapolation performance (Wang & King, 2023). Nevertheless, the pairwise approach surpassed it by incorporating better pairwise learning using a dedicated pairwise model. In standard regression, when ML algorithms learn from seen examples and try to predict unseen examples from their "experience", it can be difficult to extrapolate out of its "experience" domain. In contrast, the pairwise approach learns from feature differences, which are sometimes more generalisable than the feature values. It learns to predict the difference between training and test samples, directly aiming to determine if a test sample could win over the training samples. This transformed objective enhances the extrapolation performance of the pairwise formulation.

While using ranking algorithms like Score-based Trueskill (PA-V2), which utilises the "match scores" in the ranking process, yields a slight increase in performance, we believe that the emphasis on the correct prediction of win or loss outweighs the addition of score differences, as explained in Sect. 3.2.1 and B.3. With a slight loss in the prediction of signed differences and the extra information about absolute differences, the advantage of PA-V2 is somewhat balanced out. Overall, it is slightly better than the original Trueskill. Another advantage of PA-V2 is the convenient estimations of the y_{test} from the regressive pairwise predictions. With the material discovery datasets, we observed the reduction in MSE with PA-V2, as reported by Tynes et al. (2021).

We believe that the extrapolation ability of the pairwise approach could be employed directly to fulfil the exploitation duty in active learning (AL) tasks for top-performing samples. Tynes et al. have also revealed the advantage of a pairwise approach for uncertainty-driven AL tasks, which encourages the exploration of the wider domain by selecting samples with less confident predictions (Tynes et al., 2021). We believe it is possible to develop pairwise-approach-based AL, combining both the exploration and extrapolation traits found by their study and ours. Although their study and ours adopted two different

ways of generating pairwise features, the core intention is always to describe the difference between pairs of samples. Therefore, this diversity will make a small impact on pairwise learning, as it arises from the choice of data-preprocessing techniques (e.g. they used the one-hot encoding whereas we used the ordinal encoding).

The main limitation of the pairwise approach is the additional time and memory requirement to train a pairwise model, also as pointed out by Tynes et al. (2021), because the training set size needs to be squared for the pairwise approach. Some techniques such as batch training and sub-sampling could potentially mitigate this. More generally, improvements in computer hardware will increasingly remove this limitation. Nevertheless, the pairwise approach can be useful in novel discovery projects with a limited budget or where data is scarce to better explore the surrounding space. It was also noted that our pairwise approach can suggest some false positives when the dataset is large. How to tackle this issue will be one of the topics for our future research.

6 Conclusion

This study revealed the general applicability of the pairwise approach over thousands of datasets. We proposed a new pairwise configuration by first learning a bivariate function, $F(\text{sample 1, sample 2}) \rightarrow \Delta y$, then ranking the samples through ranking algorithms. We applied the pairwise approach to a variety of problems and datasets to receive a consistent conclusion that it is very advantageous in helping to extrapolate the target value space which can be limited by solely regression training. The pairwise approach strongly promotes the identification of extrapolating samples by almost always finding more extrapolating test samples than the standard approach. It is also better at finding top-performing test samples, by outperforming the standard approach in identifying equally or more top-performing samples on more than 70% of the datasets. The use of a score-based ranking algorithm or C3-type test pairs in ranking can slightly further boost the outperformance. It was also observed that the pairwise approach is more effective when applied to smaller datasets. The pairwise approach was applied successfully in a practical problem, the stock selection, to enable greater capital growth.

Appendix 1: Notations

Symbol	Description
f	A general supervised ML model.
f_r	A general supervised regression model.
f_c	A general supervised classification model.
F	A general supervised ML model that requires two samples' input for pairwise learning.
F_r	A general supervised regression model that requires two samples' input for pairwise learning.
$f1_{\text{extrap}}$	F1 score of a ML method to retrieve extrapolating samples
$f1_{\text{top10\%}}$	F1 score of a ML method to retrieve top 10% samples
g	The feature transformation function that generates the pairwise feature vectors.
N_f	The number of features in a dataset.
N_{train}	The number of training samples.

Symbol	Description
N_{test}	The number of test samples
p_{extrap}	Precision of a ML method to retrieve extrapolating samples
$p_{\text{top10\%}}$	Precision of a ML method to retrieve top 10% samples
r	The ranking positions of a dataset.
r_{extrap}	Recall of a ML method to retrieve extrapolating samples
$\hat{r}$	The predicted ranking positions of a dataset.
$r_{\text{top10\%}}$	Recall of a ML method to retrieve top 10% samples
R	A ranking algorithm that ranks based on absolute wins or losses.
R_s	A ranking algorithm that ranks based on match score differences.
S_A	Sample A.
$\mathbf{x}$	The feature vectors of a dataset.
$\mathbf{x}_i$	The feature vector of the ith sample in a dataset.
x	The feature vector of a single sample.
$x_{A,i}$	The ith feature of the sample A.
$\mathbf{x}_{\text{test}}$	The feature vectors of a test set.
$\mathbf{X}$	The pairwise-formulated feature vectors of a dataset.
$X_{AB,i}$	The ith pairwise feature of the pair between sample A and sample B.
y	The target values of a dataset.
y_i	The target value of the ith sample in a dataset.
y_{train}	The target values of a train set.
$y_{\text{train},i}$	The ith target value of a train set.
y_{test}	The target values of a test set.
$\hat{y}_{\text{test}}$	The predicted target values of a test set.
$\hat{y}$	The predicted target values of a dataset via a ML method.
$\hat{y}_{\text{test}}$	The predicted target values of a test set via a ML method.
$\hat{y}_{\text{test},i}$	The ith target value of a test set.
Y	The pairwise differences of the target values of a dataset.
Y_{C2}	The pairwise differences of the target values of C2-type pairs.
$Y_{C2,ij}$	The pairwise differences of the target values of C2-type pair between sample i and sample j.
$\hat{Y}$	The predicted pairwise differences of the target values of a dataset via a ML method.
Y_{AB}	The pairwise difference in target values between sample A and sample B.

Appendix 2: Additional discussion on the methods

See Figs. 7, 8, 9,10, 11 and Tables 2, 3, 4.

Pairwise Boolean features

For Boolean features, the proposed way of generating pairwise features is called ordinal encoding. It is often used for categorical features and each category value is assigned an integer value. Another popular way to encode machine-readable numerical values for categorical features is one-hot encoding. It assigns Boolean bits to describe the absence or presence of each category. Therefore, it needs to at least double the size of the feature space. In the pairwise case, one-hot encoding is equivalent to the concatenation of features

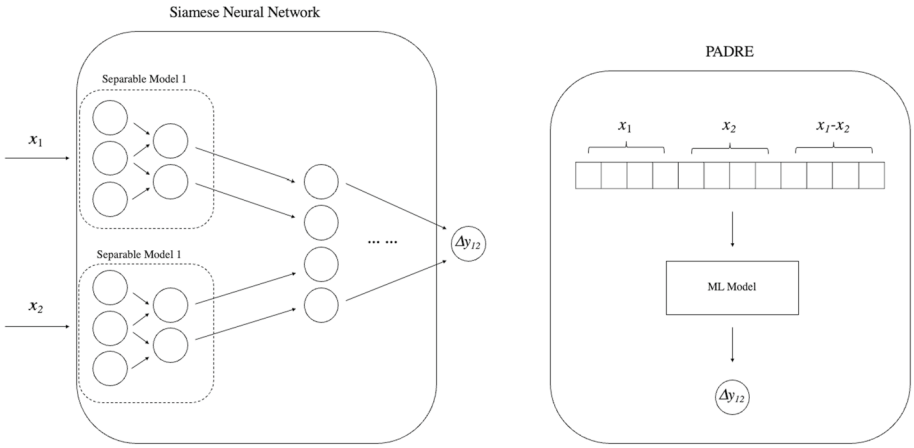


Fig. 7 Comparison between using SNN or PADRE for pairwise difference learning

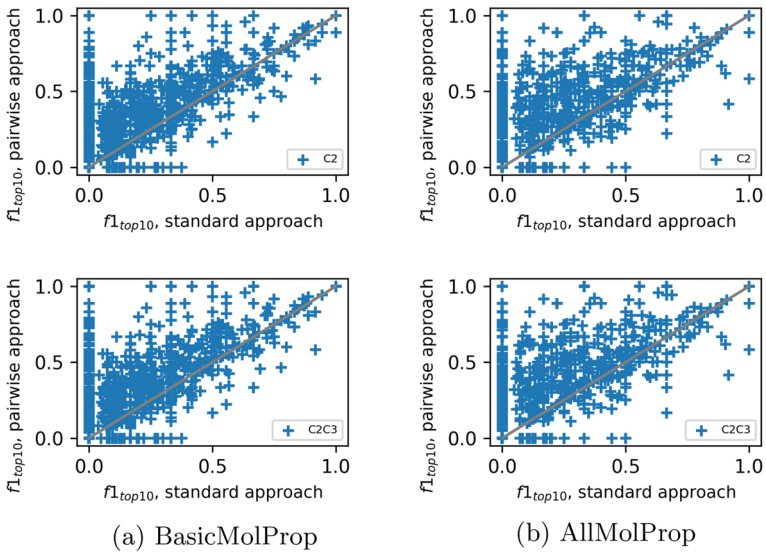


Fig. 8 The f1 score for top-performing drugs obtained by the pairwise approach versus those metrics obtained by the standard approach over **a** 1526 ChEMBL-BasicMolProp datasets, **b** 1362 ChEMBL-AllMolProp datasets, using RF

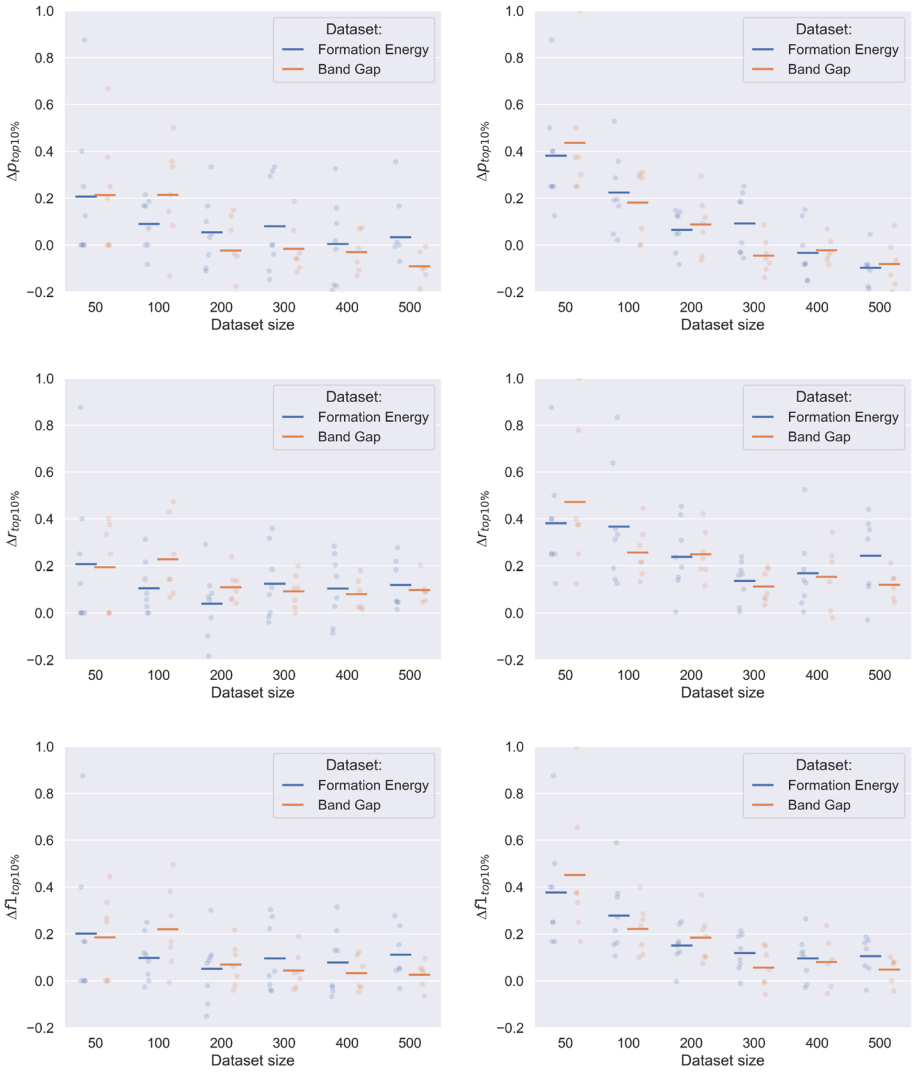
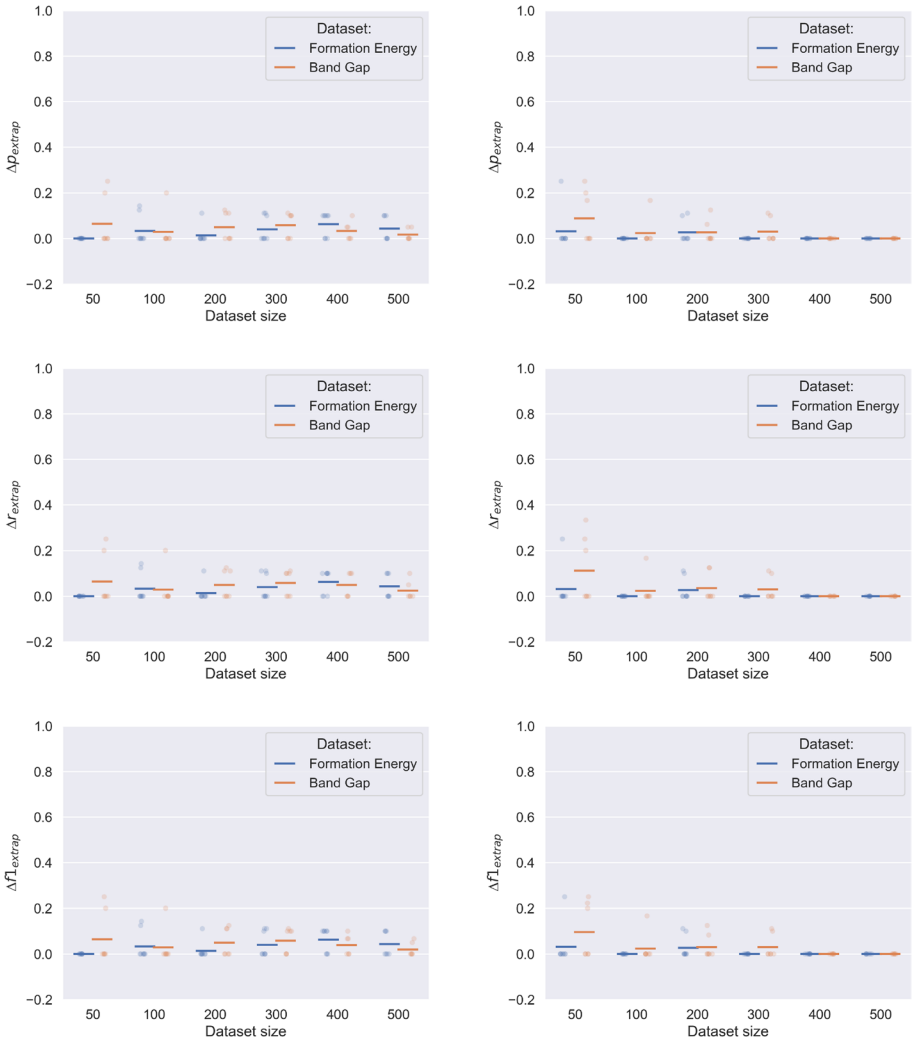


Fig. 9 The increase in precision, recall and f1 score for top-performing samples versus the size of subsets of band gap and formation energy datasets with default RF

of two samples to generate the pairwise features. Considering the large expansion of training set by permutation, the further expansion in the feature size can greatly increase training time. Furthermore, our experiments on ChEMBL datasets have shown that one-hot encoding made little difference in the training accuracy. Therefore, we decided to use ordinal encoding for the pairwise features. In ordinal encoding, the choice of the integer value for each category is not restricted. Despite potential doubts regarding the effect of their relative magnitudes under numeric transformations, it has been demonstrated not to affect



(a) PA-V1-C2 (b) PA-V2-C2

Fig. 10 The increase in precision, recall and f1 score for extrapolating samples versus the size of subsets of band gap and formation energy datasets with default RF

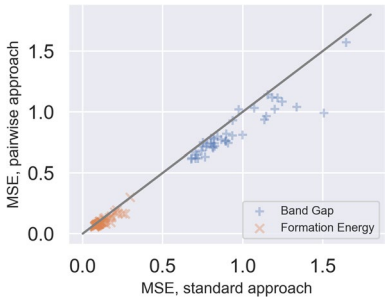


Fig. 11 MSE of the predicted y_{test} using the estimations from the regressive pairwise predictions of the pairwise approach (PA-V2-C2) versus the direct predictions of the standard approach

Table 2 The percentage of datasets showing the pairwise approach with Elo's rating algorithm and other arrangements: (i) had an equal or better performance than the standard approach, i.e., metric(pairwise)>= metric(standard), (ii) was better than the standard approach, i.e., metric(pairwise) > metric (standard), excluding datasets showing equal performance

Datasets	Boolean ChEMBL 1436 datasets			
Percentage type	(i)		(ii)	
PA arrangement	Elo-C2	Elo-C2C3	Elo-C2	Elo-C2C3
ML Methods	RF			
p_{extrap}	100.0%	100.0%	100.0%	100.0%
r_{extrap}	100.0%	100.0%	100.0%	100.0%
$f1_{extrap}$	100.0%	100.0%	100.0%	100.0%
$p_{top10\%}$	71.8%	70.1%	70.0%	68.5%
$r_{top10\%}$	97.9%	98.0%	97.8%	97.9%
$f1_{top10\%}$	84.0%	83.1%	83.2%	82.3%

All the values have a binomial p value < 0.05

Table 3 Stock ratios used as attributes in the stock selection model

Ratios	Description
PE-ratio	Price-to-earnings ratio = share priceearnings per share
PB-ratio	Price-to-book ratio = share price/book value per share
PS-ratio	Price-to-sales ratio = share price/sales per share
ROE	Return on equity (after tax) = net income after tax/shareholders' equity
ROA	Return on asset (after tax) = net income after tax/total assets
OPM	Operating profit margin = operating income/net sales
NPM	Net profit margin = net income after tax/net sales
DE-ratio	Debt-to-equity ratio = total liabilities/shareholders' equity
CR	Current ratio = current assets/current liabilities
QR	Quick ratio = quick assets/current liabilities
ITR	Inventory turnover rate = cost of goods sold/average inventory
RTR	Receivables turnover rate = net credit sales/average accounts receivable
OIG	Operating income growth rate = (operating income at the current year - operating income at the previous year) / operating income at the previous year
NIG	Net income growth rate = (net income after tax at the current year - net income after tax at the previous year) / net income after tax at the previous year

Table 4 The hyperparameters tuned in the stock selection experiments

Model	Hyperparameters
SVM (Gaussian radial basis function)	C, regularisation parameter
	gamma, kernel coefficient
RF	max_features, the maximum number of features
	max_samples, the maximum number of samples
	n_estimators, the maximum number of estimators

our study through simple tests. We endeavoured to assign each combination listed above with a different value (e.g., $x_{A,i} = 1, x_{B,i} = 1 \rightarrow X_{AB,i} = -1$; $x_{A,i} = 0, x_{B,i} = 1 \rightarrow X_{AB,i} = 0$). We have also tried a different set of ordinal values, for example, using 1, 2, 3, 4 instead of -1, 0, 1, 2. In both tests, the results were hardly varied by the choice of ordinal values.

Pairwise regressive features

For regressive feature vectors, instead of concatenating $x_{A,i}$, $x_{B,i}$ and $(x_{A,i} - x_{B,i})$ all together as used in PADRE, we only concatenated the difference vector and one of the constitutive vector. We believe that the information in Pair AB $(x_{A,i} - x_{B,i}) \oplus x_{A,i} \oplus x_{B,i})$ is somewhat repetitive to Pair BA $(x_{B,i} - x_{A,i}) \oplus x_{B,i} \oplus x_{A,i})$, and that the simplified version, $(x_{A,i} - x_{B,i}) \oplus x_{A,i}$, is sufficient to represent the Pair AB and distinguish it from Pair BA. Most importantly, the expansion in the feature dimension can be minimised by a third compared to PADRE, which can be significant for datasets with a large number of features.

We investigated the choice of ranking algorithm. We experimentally examined several generic ranking algorithms and found that the choice of the generic ranking algorithm can affect the ranking accuracy given the same sets of $\text{sign}(Y)^{\text{pred}}$, usually by about 1%. It is believed that the main contribution to accurate ranking should come from the accuracy in $\text{sign}(Y)^{\text{pred}}$ rather than the rating algorithm. Therefore, Trueskill was selected and used to rank the samples from the predicted signs. Trueskill was originally designed to rank players in the game "Halo" (Herbrich et al., 2007). Because it assumes variances both in players' performance and skill levels, it can deal with potential conflicts in match outcomes, in our case, conflicts in $\text{sign}(Y)^{\text{pred}}$ due to learning errors. For example, when $\text{sign}(Y_{AB})^{\text{pred}} = -1$ and $\text{sign}(Y_{BC})^{\text{pred}} = -1$, it implies that sample A < sample B < sample C. But if $\text{sign}(Y_{AC})^{\text{pred}} = 1$, which implies sample A > sample C, then these predictions suggest opposite opinions. This situation is similar to game tournaments, in which a strong player does not necessarily win every time.

Trueskill vs. score-based trueskill

With the use of ranking algorithms such as Score-based Trueskill (PA-V2) where the "match scores" participate as a part of the ranking process, we only observed a small increase in performance, despite the extra information about how much two samples differ in a pairwise comparison. One explanation is that if we only train on the classification model on the signed differences of pairs, we can obtain better accuracy of the predicted signs which is better for ranking. Note that the effect of predicting win or loss is more important for correct ranking compared to the effect of the score differences. Considering a pairwise comparison between Sample A and Sample B which has a small difference in y between them, say $Y_{AB} = 0.10$. For a regression model, the loss function is based on the mean squared error (MSE). So predictions of $Y_{AB} = 0.31$ and $Y_{AB} = -0.11$ are treated in the same way. However, this change in sign can make a greater impact on the ranking of these two samples. A wrong sign prediction will penalise more ranking scores of Sample A compared to an off prediction on the absolute difference. So the ranking update using $Y_{AB} = 0.31$ is closer to the true ranking compared to the ranking update using $Y_{AB} = -0.01$ where the sign is wrong but the absolute difference is better predicted. We have seen in experiments that the accuracy of the signed difference predicted by a regression model is poorer compared to a classification model. Combining this with the effect of more accurate ranking with absolute differences (when the sign is predicted correctly), the advantage of

PA-V2 is slightly balanced out. But overall, we can still see a slight increase in performance with PA-V2.

Datasets

ChEMBL

ChEMBL is a chemical database of bioactive molecules (Mendez et al., 2019; Olier et al., 2021). It contains a large number of molecules and their measured activities against a variety of targets. Due to their size and scope, these datasets are suitable for benchmarking ML applications in the realm of QSAR. ChEMBL features a number of different activities, in this study we are employing pXC50 as our target values, i.e. -log(measured activity). The structure of drug molecules is represented by the commonly employed Morgan fingerprint (1024 bits, radius=2) encoding the molecular substructures by Boolean values.

Gene expression datasets

The human gene expression datasets (accession code GSE70138) from the Library of Integrated Network-based Cellular Signatures data (LINCS) (Koleti et al., 2018). This set of datasets contains the measured gene expression level across different tissue types and drug treatments in cancer cell lines. There are a total of 978 human genes, each of which was measured under 118,050 experimental conditions. Each dataset is the expression levels of a gene, measured and processed as level 5 differential gene expression signatures, under a series of conditions. The conditions are featured into 1,154 Boolean values describing drugs' fingerprints (1024 bits) and experimental settings, which include 83 dosages, 14 cell types and 3 time points.

Material discovery datasets

Formation energy and band gap were selected as the properties to be studied here for their large amount of available data since they have been widely studied. They collected the data from the Materials Project database which contains 83,989 compounds. The dataset was condensed into a representative set of 10042 after data filtering removed duplicated composition, single-element composition, ill-converged samples, and uncommon elements. Regarding the compound representation, we concatenated the two 1D representations used in Xiong et al.'s study, Magpie (Materials Agnostic Platform for Informatics and Exploration) and element one-hot composition representation. The former computes continuous feature values for a given material including elemental property statistics of 22 different elemental properties. The latter describes the presence and absence of the 52 elements in a compound, weighted by the composition ratio per element.

Acknowledgements This work was partially supported by the Wallenberg AI, Autonomous Systems and Software Program (WASP) funded by the Alice Wallenberg Foundation. Funding was also provided by the Chalmers AI Research Centre and the UK Engineering and Physical Sciences Research Council (EPSRC) grant nos: EP/R022925/2 and EP/W004801/1.

Author Contributions Y.W. and R.D.K. proposed the idea and designed the experiments. Y.W. did the experiments and analysis, wrote the manuscript and prepared the figures. All authors reviewed the manuscript.

Code & data availability The code and data to reproduce the results can be found: https://github.com/iris-ywang/pairwise_formulation_experiments

Declarations

Conflict of interest The authors declare that they have no Conflict of interest.

Ethical approval N/A

Consent for publication N/A

Consent for participate N/A

References

Agarwal, S., Dugar, D., & Sengupta, S. (2010). Ranking chemical structures for drug discovery: a new machine learning approach. *Journal of Chemical Information and Modeling, 50*(5), 716–731. https://doi.org/10.1021/ci9003865. Publisher: American Chemical Society. Accessed 2023-03-25.

Al-Dabbagh, M. M., Salim, N., Himmat, M., Ahmed, A., & Saeed, F. (2017). Quantum probability ranking principle for ligand-based virtual screening. *Journal of Computer-Aided Molecular Design, 31*(4), 365–378. https://doi.org/10.1007/s10822-016-0003-4

Bao, H., Niu, G., & Sugiyama, M. (2018). Classification from pairwise similarity and unlabeled data. In: Dy, J., & Krause, A. (eds.) *Proceedings of the 35th International Conference on Machine Learning. Proceedings of Machine Learning Research*, vol. 80, pp. 452–461. PMLR, Tokyo, https://proceedings.mlr.press/v80/bao18a.html

Cramer, R. D. (2012). The inevitable QSAR renaissance. *Journal of Computer-Aided Molecular Design, 26*(1), 35–38. https://doi.org/10.1007/s10822-011-9495-0. Accessed 2023-03-25.

Fernández-Llaneza, D., Ulander, S., Gogishvili, D., Nittinger, E., Zhao, H., & Tyrchan, C. (2021). Siamese recurrent neural network with a self-attention mechanism for bioactivity prediction. *ACS Omega, 6*(16), 11086–11094. https://doi.org/10.1021/acsomega.1c01266. Publisher: American Chemical Society. Accessed 2023-12-22.

Fralish, Z., Chen, A., Skaluba, P., & Reker, D. (2023). DeepDelta: Predicting ADMET improvements of molecular derivatives with deep learning. *Journal of Cheminformatics, 15*(1), 101. https://doi.org/10.1186/s13321-023-00769-x. Accessed 2023-12-05.

Guo, S., Sanner, S., Graepel, T., & Buntine, W. (2012). Score-Based Bayesian Skill Learning. In P. A. Flach, T. De Bie, & N. Cristianini (Eds.), *Machine Learning and Knowledge Discovery in Databases. Lecture Notes in Computer Science* (pp. 106–121). Berlin: Springer. https://doi.org/10.1007/978-3-642-33460-3_12

Herbrich, R., Minka, T., & Graepel, T. (2007). TrueSkill(TM). A Bayesian Skill Rating System, pp. 569–576. https://www.microsoft.com/en-us/research/publication/trueskilltm-a-bayesian-skill-rating-system/ Accessed 25-Apr-2023

Huang, C. F. (2012). A hybrid stock selection model using genetic algorithms and support vector regression. *Applied Soft Computing, 12*(2), 807–818. https://doi.org/10.1016/j.asoc.2011.10.009. Accessed 2023-09-26.

Jiménez-Luna, J., Pérez-Benito, L., Martínez-Rosell, G., Sciabola, S., Torella, R., Tresadern, G., & Fabritiis, G. D. (2019). DeltaDelta neural networks for lead optimization of small molecule potency. *Chemical Science, 10*(47), 10911–10918. https://doi.org/10.1039/C9SC04606B. Publisher: The Royal Society of Chemistry. Accessed 2023-12-22.

Kauwe, S. K., Graser, J., Murdock, R., & Sparks, T. D. (2020). Can machine learning find extraordinary materials? *Computational Materials Science, 174*, 109498. https://doi.org/10.1016/j.commatsci.2019.109498. Accessed 2022-10-02.

King, R. D., Orhobor, O. I., & Taylor, C. C. (2021). Cross-validation is safe to use. *Nature Machine Intelligence, 3*(4), 276–276. https://doi.org/10.1038/s42256-021-00332-z. Number: 4 Publisher: Nature Publishing Group. Accessed 2022-11-03.

Koleti, A., Terryn, R., Stathias, V., Chung, C., Cooper, D. J., Turner, J. P., Vidovic, D., Forlin, M., Kelley, T. T., D'Urso, A., Allen, B. K., Torre, D., Jagodnik, K. M., Wang, L., Jenkins, S. L., Mader, C., Niu, W., Fazel, M., Mahi, N., … Schürer, S. C. (2018). Data Portal for the Library of Integrated Network-based Cellular Signatures (LINCS) program: integrated access to diverse large-scale cellular perturbation response data. *Nucleic Acids Research, 46*(D1), 558–566. https://doi.org/10.1093/nar/gkx1063

Korff, M., & Sander, T. (2022). Limits of Prediction for Machine Learning in Drug Discovery. *Frontiers in Pharmacology, 13*, 832120. https://doi.org/10.3389/fphar.2022.832120

Lehmann, R., & Wohlrabe, K. (2017). An elo ranking for economics journals. *Economics Bulletin, 37*, 2282–2291.

Liu, J., & Ning, X. (2017). Multi-assay-based compound prioritization via assistance utilization: A machine learning framework. *Journal of Chemical Information and Modeling, 57*(3), 484–498. https://doi.org/10.1021/acs.jcim.6b00737. Publisher: American Chemical Society. Accessed 2023-05-19.

Macrotrends | The Long Term Perspective on Markets. https://www.macrotrends.net Accessed 15-Dec-2023

Mauri, A., Consonni, V., Pavan, M., Todeschini, R., & Chemometrics, M. (2006). Dragon software: An easy approach to molecular descriptor calculations. *Match, 56*(2), 237–248.

McNutt, A. T., & Koes, D. R. (2022). Improving $\delta\delta$ G Predictions with a Multitask Convolutional Siamese Network. *Journal of Chemical Information and Modeling, 62*(8), 1819–1829. https://doi.org/10.1021/acs.jcim.1c01497. Publisher: American Chemical Society. Accessed 2023-12-22.

Mendez, D., Gaulton, A., Bento, A. P., Chambers, J., De Veij, M., Félix, E., Magariños, M., Mosquera, J., Mutowo, P., Nowotka, M., Gordillo-Marañón, M., Hunter, F., Junco, L., Mugumbate, G., Rodriguez-Lopez, M., Atkinson, F., Bosc, N., Radoux, C., Segura-Cabrera, A., … Leach, A. (2019). ChEMBL: Towards direct deposition of bioassay data. *Nucleic Acids Research, 47*(D1), 930–940. https://doi.org/10.1093/nar/gky1075. Accessed 2023-03-25.

Meredig, B., Antono, E., Church, C., Hutchinson, M., Ling, J., Paradiso, S., Blaiszik, B., Foster, I., Gibbons, B., Hattrick-Simpers, J., Mehta, A., & Ward, L. (2018). Can machine learning identify the next high-temperature superconductor? Examining extrapolation performance for materials discovery. *Molecular Systems Design & Engineering, 3*(5), 819–825. https://doi.org/10.1039/C8ME00012C. Publisher: Royal Society of Chemistry. Accessed 2023-02-10.

Nicolotti, O. (Ed.). (2018). *Computational Toxicology: Methods and Protocols. Method in Molecular Biology,* (Vol. 1800). New York: Springer. https://doi.org/10.1007/978-1-4939-7899-1

Olier, I., Orhobor, O. I., Dash, T., Davis, A. M., Soldatova, L. N., Vanschoren, J., & King, R. D. (2021). Transformational machine learning: Learning how to learn from many related scientific problems. *Proceedings of the National Academy of Sciences, 118*(49), 2108013118. https://doi.org/10.1073/pnas.2108013118

Olier, I., Sadawi, N., Bickerton, G. R., Vanschoren, J., Grosan, C., Soldatova, L., & King, R. D. (2018). Meta-QSAR: A large-scale application of meta-learning to drug design and discovery. *Machine Learning, 107*(1), 285–311. https://doi.org/10.1007/s10994-017-5685-x. Accessed 2023-01-29.

Park, Y., & Marcotte, E. M. (2012). Flaws in evaluation schemes for pair-input computational predictions. *Nature Methods, 9*(12), 1134–1136. https://doi.org/10.1038/nmeth.2259. Accessed 2022-07-14.

Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., Blondel, M., Prettenhofer, P., Weiss, R., Dubourg, V., Vanderplas, J., Passos, A., Cournapeau, D., Brucher, M., Perrot, M., & Duchesnay, E. (2011). Scikit-learn: Machine learning in python. *Journal of Machine Learning Research, 12*(85), 2825–2830. Accessed 2023-03-25.

Rathke, F., Hansen, K., Brefeld, U., & Müller, K.-R. (2011). StructRank: A new approach for ligand-based virtual screening. *Journal of Chemical Information and Modeling, 51*(1), 83–92. https://doi.org/10.1021/ci100308f. Accessed 2022-11-03.

Tong, W., Hong, H., Xie, Q., Shi, L., Fang, H., & Perkins, R. (2005). Assessing QSAR Limitations - A Regulatory Perspective. *Current Computer-Aided Drug Design, 1*(2), 195–205.

TrueSkill - trueskill 0.4.5 documentation. https://trueskill.org/ Accessed 25-Apr-2023

Tynes, M., Gao, W., Burrill, D. J., Batista, E. R., Perez, D., Yang, P., & Lubbers, N. (2021). Pairwise difference regression: A machine learning meta-algorithm for improved prediction and uncertainty quantification in chemical search. *Journal of Chemical Information and Modeling, 61*(8), 3846–3857. https://doi.org/10.1021/acs.jcim.1c00670. Accessed 2022-07-14.

Tyrchan, C., & Evertsson, E. (2017). Matched molecular pair analysis in short: Algorithms, applications and limitations. *Computational and Structural Biotechnology Journal, 15*, 86–90. https://doi.org/10.1016/j.csbj.2016.12.003. Accessed 2022-07-24.

Wang, Y., & King, R. D. (2023). Extrapolation is Not the Same as Interpolation. In A. Bifet, A. C. Lorena, R. P. Ribeiro, J. Gama, & P. H. Abreu (Eds.), *Discovery Science. Lecture Notes in Computer Science* (pp. 277–292). Cham: Springer. https://doi.org/10.1007/978-3-031-45275-8_19

Watson, O. P., Cortes-Ciriano, I., Taylor, A. R., & Watson, J. A. (2019). A decision-theoretic approach to the evaluation of machine learning algorithms in computational drug discovery. *Bioinformatics, 35*(22), 4656–4663. https://doi.org/10.1093/bioinformatics/btz293. Accessed 2022-10-07.

Wetzel, S. J., Ryczko, K., Melko, R. G., & Tamblyn, I. (2022). Twin neural network regression. *Applied AI Letters, 3*(4), 78. https://doi.org/10.1002/ail2.78https://onlinelibrary.wiley.com/doi/pdf/10.1002/ail2.78.

Xiong, Z., Cui, Y., Liu, Z., Zhao, Y., Hu, M., & Hu, J. (2020). Evaluating explorative prediction power of machine learning algorithms for materials discovery using k -fold forward cross-validation. *Computational Materials Science, 171*, 109203. https://doi.org/10.1016/j.commatsci.2019.109203. Accessed 2022-09-11.

Yu, J., Li, Z., Chen, G., Kong, X., Hu, J., Wang, D., Cao, D., Li, Y., Huo, R., Wang, G., Liu, X., Jiang, H., Li, X., Luo, X., & Zheng, M. (2023). Computing the relative binding affinity of ligands based on a pairwise binding comparison network. *Nature Computational Science, 3*(10), 860–872. https://doi.org/10.1038/s43588-023-00529-9. Number: 10 Publisher: Nature Publishing Group. Accessed 2023-10-28.

Zhang, W., Ji, L., Chen, Y., Tang, K., Wang, H., Zhu, R., Jia, W., Cao, Z., & Liu, Q. (2015). When drug discovery meets web search: Learning to Rank for ligand-based virtual screening. *Journal of Cheminformatics, 7*(1), 5. https://doi.org/10.1186/s13321-015-0052-z. Accessed 2022-07-14.