**Large-Scale Prediction of Activity Cliffs Using Machine and Deep Learning Methods of Increasing Complexity**

Shunsuke Tamura,1,2 Tomoyuki Miyao,2 and Jürgen Bajorath1\*

2Data Science Center and Graduate School of Science and Technology, Nara Institute of Science and Technology, 8916-5 Takayama-cho, Ikoma, Nara, 630-0192, Japan.

1Department of Life Science Informatics, B-IT, LIMES Program Unit Chemical Biology and Medicinal Chemistry, Friedrich-Hirzebruch-Allee 5/6, Rheinische Friedrich-Wilhelms-Universität, D-53115 Bonn, Germany.

\*To whom correspondence should be addressed:

Tel: +49-228-7369-100, Fax: +49-228-7369-101, E-mail: [bajorath@bit.uni-bonn.de](mailto:bajorath@bit.uni-bonn.de)

**Abstract**

**1. Introduction**

Activity cliffs (ACs) were originally introduced as structural similar compounds with large differences in activity (against the same target), presenting major problems for standard quantitative structure-activity relationship (QSAR) predictions [1]. ACs are of high relevance for medicinal chemistry, given that they capture small chemical modifications with large consequences for specific biological activities [2]. As such, ACs represent extreme examples of SAR discontinuity, which is encountered during compound optimization and might –or might not– be desirable, depending on the development stage of compound series [2]. In medicinal chemistry, ACs are best rationalized as structural analogues (belonging to the same series) having large potency differences. However, for AC definition and assessment, a variety of molecular similarity and potency difference criteria have been introduced [2] and the AC concept has been further refined over time, both from a medicinal chemistry and chemoinformatics perspective [3].

While ACs were systematically explored on the basis of compound activity data analysis [2], which yielded most current insights into ACs and their distribution across different compound classes [2,3], various attempts have also been made to computationally predict ACs [3]. Compared to predictions of compound activity (or other properties) using QSAR modeling or machine learning (ML) AC predictions were principally challenging because they needed to focus on compound pairs, rather than individual molecules, which required methodological adjustments and extensions. First attempts to predict individual AC compounds or complete ACs were made a decade ago using random forest (RF) and support vector machine modeling (SVM), respectively [4,5]. In these original studies, SVM predictions using matched molecular pair (MMP) representations of ACs and especially designed MMP kernels to facilitate predictions at the level of compound pairs yielded surprisingly high prediction accuracy (80-90%). Therefore, the SVM-MMP formalism was also applied in subsequent studies to further explore and refine AC predictions [6,7]. As a simper alternative to the use of MMP kernels, the condensed graph of reaction (CGR) formalism was also applied to represent MMPs for AC predictions using different QSAR/ML methods, reaching an accuracy overall comparable to SVM [8]. For representing individual compounds or MMPs, standard fingerprint descriptors from chemoinformatics, i.e., bit string representations of chemical structure, were used in these studies. Following a different approach, ACs were also predicted on the basis of target-bound compound conformations and three-dimensional (3D) binding mode similarity measures [9], yielding lower accuracy than SVM modeling. Recently, deep learning (DL) has been applied to predict ACs from MMP images using convolutional neural networks [10,11] or from molecular graphs involving representation learning with graph neural networks (GNNs) [12]. These DL approaches to AC prediction reached similarly high prediction accuracy as earlier ML studies (for example, with area under the receiver-operating characteristic curve (AUC) values greater 0.9). In addition to classification models for AC prediction, regression models have also been applied to predict the potency of individual AC compounds [13,14]-

All AC prediction efforts reported over time applied a general 100-fold difference in compound potency as a criterion for AC definition, irrespective of the compound classes under investigation. Furthermore, with the exception of 3D AC predictions [9], these studies consistently applied the MMP formalism as a similarity criterion for AC definition and representation. Moreover, all of these studies also had in common that they reported AC predictions only for a limited number of compound activity classes; always fewer than 10 and in some cases –including the DL investigations– only two or three. Since the system set-up, compound classes, and calculation conditions largely varied in the studies, they can also not be rigorously compared.

In this work, we report a first large-scale AC prediction over 100 compound activity classes using machine learning methods of increasing complexity (including DL). By design, this study aimed to enable a direct comparison of various methodologies for AC prediction and provide a general assessment of prediction accuracy across many different compound classes. Furthermore, different from earlier studies, ACs were defined and predicted on the basis of statistically significant activity class-dependent potency differences, derived from class-specific compound potency distributions.

**2. Methods**

**2.1. Compound data sets**

Compounds activity classes were extracted from ChEMBL database (version 29) 1 based on the following criteria: molecular mass less than 1000 Da, target confidence score of 9, interaction relationship type ‘D’, and a numerically exact potency value. Only Ki or Kd measurements were considered. Each activity class consisted of qualifying compounds with reported activity against an individual target. In addition to the compound-based criteria given above, activity classes were required to meet AC analysis criteria, as specified below. A total of 100 activity classes were assembled. Their targets and composition are reported in **Supplementary Table S1**.

**2.2. Activity cliff definition**

**2.2.1. Structural similarity criterion**

As an intuitive representation of the structural similarity for small chemical modification, matched molecular pair (MMP) formalism was induced. An MMP was formed by a pair of compounds that share a common core structure and have their unique substituents at a single site on the core. ACs are well described by using MMPs, termed as MMP-cliffs, to represent the potency change derived from the corresponding chemical modification. An MMP-cliffs 2 is defined by a pair of structurally similar compounds forming an MMP with significant difference in potency. MMPs were generated with computationally efficient algorithm using Hussain and Rea approach 3 implemented in the previous study 4. For MMP generation, substructure exchange was limited in no more than 13 heavy atoms and maximum difference between substituent of a compound was limited in no more than eight heavy atoms. After MMP generation, MMPs having small core with less than 10 heavy atoms were discarded.

### 2.2.2. Activity class-dependent potency difference criteria

Initially, MMP-cliffs were qualified as ACs if the MMPs have greater than 100-folds difference in potency (∆*pKi* > 2)5. However, in previous study, activity class-dependent ACs were observed and activity class-dependent criteria defined from statistical meaningful distribution should be considered6. Following previous study, the threshold for an activity class was calculated by mean of the potency difference distribution added two standard deviations.

MMPs with potency difference of greater than the criteria were labelled as ACs. To prevent boundary effect derived from potency difference, only MMPs with less than 10-folds difference in potency (∆*pKi* < 1) were labelled as non-AC. The rest of MMPs with intermediate potency difference were discarded. Activity classes used in the further analysis were selected with two distinct criteria corresponds to the data splitting approach mentioned later.

### 2.2.3. Data leakage phenomena

Some MMPs in a data set for an activity class can share compounds and the similarity of these MMPs are much higher than the other pair of MMPs. When a data set for an activity class is randomly separated into training and test set, these MMPs can appear in the both set simultaneously. This could cause that models do not predict based on the features related to AC forming but much focus on the features in shared compounds, which termed data leakage phenomena in present study. Data leakage phenomena also artificially increase the model performance in AC prediction 7. To consider this phenomena, two data separating approach were conducted. In ‘data leakage possibly included’ split, data sets were randomly separated into training set (80%) and test set (20%). In ‘data leakage excluded’ split, advanced cross-validation (AXV) approach proposed by Horvath et al 8. was referenced. First, for each activity class, compounds used for making MMPs were collected and several compounds were randomly selected as ‘kept-out’ pool. Then, MMPs were recursively selected and assigned to training or test set based on following criteria. If neither compound of the MMP was shared with kept-out pool, the MMP was assigned to training set. If both compounds were in kept-out pool, the MMP was assigned to external test set. If one compound was in kept-out pool, the MMP was no longer used for the further analysis.

For ‘data leakage possibly included’ set, activity classes with at least 50 MMPs and having at least one MMP-cliff and MMP-noncliff were used. For ‘data leakage excluded’ set, activity classes with at least 20 MMP-cliff were used, which resulted in 42 activity classes. A total of 100 and 42 activity classes were assembled for ‘data leakage possibly included’ set and ‘data leakage excluded’ set, respectively.

### 2.2.4. Molecular representation

MMP fingerprints 9 were used to represent an MMP as a single feature vector. MMP fingerprints is assembled by concatenating individual fingerprints for MMP core and a pair of substituents. Extended connectivity fingerprints of bond diameter 4 (ECFP4) 10 were used as fingerprints algorithm. During ECFP4 calculation, features within bond diameter of one were eliminated from ECFP feature collection to clarify the contributions of features over bond diameter of two. For each part, identifiers corresponding to the features were sorted in ascending order and assigned to bits in the fingerprints in the same order. This is performed to prevent feature collision and make features contribute to AC prediction as many as possible. More to focus on the chemical transformation between compounds in an MMP, two fingerprints with the dimension same as the fingerprints for the substituents were generated. One was made by setting bits to one if the bits were one in both of the fingerprints, and the other if the bits were one in either of the fingerprints. This clarifies the contribution of common and unique features between substituents. After generating individual fingerprints, the fingerprints for core, unique features of substituents, and common features of substituents were concatenated into one vector. MMP fingerprints calculation was conducted with inhouse Java and Python scripts based on *OEChem toolkit* 11.

## 2.3. Condensed graph of reaction representation

To apply graph neural network approach to AC prediction, An MMP was represented in a single graph using the condensed graph of reaction (CGR) approach 8,12. The CGR formalism was originally conceived to combine reactants and products graphs based on a superposition of invariant parts. The resulting CGR forms a completely connected graph in which each node represents an atom and each edge a bond. In a CGR, the shared core of an MMP and the two exchanged substituent fragments form a single pseudo-molecule. The subgraph coming from the low potent compound was connected with the core via a single bond and the high potent compound a hypothetical zero-order bond. An MMP was converted into a pseudo-molecule using an in-house Python script with the *RDKit* API 13.

## 2.4. Machine learning

To find best approach for AC prediction, four fingerprints-based approaches which were support vector machine (SVM), extreme gradient boosting (XGboost), random forest, and fully connected neural network (FCNN), and a graph-based approach which was message passing neural network (MPNN) were used. For FCNN and MPNN, two distinct models which differ at molecular representation for the input were build. As control calculations, *k*-nearest neighbor (kNN) approach was also compared including 1NN and 5NN. In kNN calculation, the similarity was evaluated with MMPkernel mentioned in section 2.4.1. Model constructions were performed three times for each activity class and each machine learning models were trained with three-fold internal cross-validation to optimize hyperparameters. The three pairs of training set and external test set were common among machine learning models using same seeds. The performances of the machine learning models were calculated by taking average of three trials.

All models except the two neural networks were implemented with scikit-learn 14 and the neural newtworks *PyTorch* 15. Hyperparameters of models listed in **Supplementary Table S2** were optimized with *Optuna* library 16. The hyperparameter search with *Optuna* for each model was performed for 100 times. The rest of parameters not listed in **Supplementally Table S2** except C in SVM were set to their default.

### 2.4.1. Support vector machine

SVM is a supervised learning method that aims to define the hyperplane separating given training instances with two class labels with maximizing the margin from the hyperplane 17. SVM was originally developed as linear classification method. If linear classification is not possible, SVM can be easily extend to nonlinear classification using kernel functions. In present study, MMPkernel 9 which is a product of two individual Tanimoto kernel 18 calculating core-wise and substituent-wise similarity was used as the kernel function. The parameter ‘class\_weight’ was set to ‘balanced’. The hyperparameter C was selected from a value from to divided into equal 10 steps in leaner log space with grid search.

### 2.4.2. Random forest

RF is a supervised learning method that is an ensemble of multiple decision trees generated from randomly chosen training instances using bootstrapping 19. RF predict for test instances based on the majority of the individual prediction of decision trees. The parameter ‘class\_weight’ was set to ‘balanced’.

### 2.4.3. Extreme gradient boosting

XGboost is a supervised learning method that is also ensemble of decision trees using gradient boosting 20. Gradient boosting iteratively generates decision trees so that each decision trees minimize the residual error from previous models. XGBoost is a computationally efficient and accurate extension of gradient boosting, which is achieved by parallelizing the decision tree construction.

### 2.4.4. Fully-connected neural network

A FCNN consists of a series of connected perceptrons stored in several layers. Each perceptron receives signals from previous layer transform into scaler value with activation function and send it to next layer as a signal. In this study, two distinct FCNNs were implemented based on their input; FCNN with single feature vector represented by MMPfingerprints (simply called as FCNN), FCNN with three separated ECFP4 representing core and two substituents (FCNN\_sep). For FCNN, input MMPfingerprints were mapped into probability indicating how likely the MMP forms AC. The number of nodes in hidden layers was monotonically reduced. In FCNN\_sep, the individual fingerprints were input to several hidden layers and the outputs were concatenated into a vector, which was sent to following hidden layers and transformed into the probability with softmax layer. The number of nodes in hidden layers for both individual substructures and concatenated feature vectors were also monotonically reduced. Rectified Linear Unit (ReLU) 21 was used as activation function except final layer. Binary cross-entropy with balance factor weighted by the number of positive samples was used as loss function and the loss was optimized by the Adam optimizer 22. This learning rate was managed by optim.lr\_scheduler.StepLR scheduler in PyTorch. For the scheduler, the parameter gamma was set to 0.1, and the step size, which is defined by dividing the number of training iterations by the number of steps, is a hyperparameter. The batch size was set to 128 if the number of MMPs in a training set was greater than 128; otherwise, it was equaled to the size of training size. Training steps were performed for 50 epochs during the hyperparameter search and for 100 epochs during best parameter fitting.

### 2.4.5. Message passing neural network

MPNN is a graph convolutional neural network approach which accepts molecular graph as the input and can learn the way to convert input molecular graphs to its optimal feature vectors. During MPNN training step, feature vector on each atom iteratively merged with information from its neighbor atoms and bonds so as to minimize the loss function. The initial features on each atom and bonds are listed in **Supplementally Table 3**. The transformed feature vectors of each atom are merged into single vector and connected to fully-connected neural network with several hidden layer whose output was the probability. In present study, the MPNN architecture referenced previous study 23 originally proposed by Tang et al 24. Same as FCNN and FCNN\_sep, two distinct MPNN were implemented corresponding to their input; MPNN with single CGR (simply called as MPNN), MPNN with three separated subgraphs representing core and two substituents (MPNN\_sep). In MPNN\_sep, feature vectors for each substructure were calculated with individual MPNN and concatenated into single vector which connected to fully-connected neural network. Activation function, loss function, optimizer, scheduler of optimizer, batch size, epochs, and the number of hyperparameter search were set to same setup as FCNN.

## 2.5. Performance measures

To evaluate performance of the models, balanced accuracy (BA) 25, recall, precision, and Matthew’s correlation coefficient (MCC) 26 were calculated. The four measures are defined below,

where TP, TN, FP, and FN stand for true positives, true negatives, false positives, and false negatives, respectively.

# **3. Results and Discussion**

3.1. Global performance comparison

The performance distributions of nine machine learning methods using 100 activity classes are shown in **Figure 1**. Comparing the distributions of BAs and MCCs, SVM, RF, and FCNN\_sep showed slightly better performance followed by 1NN and 5NN and both MPNN model did not show as high performance as the others. This indicates that classical fingerprints-based models are enough to build predictive models and modern GCN approaches are not required in AC prediction. Considering the input styles of neural networks, neural networks with multiple input showed higher performance than concatenated input. This result was consistent with previous findings that, in AC prediction using SVM, MMP kernel which calculated similarity of core and substituents separately showed better performance than Tanimoto kernel with concatenated fingerprints.

Exemplary MMPs which were predicted accurately with high (MMP-cliff) or low (MMP-noncliff) probability are depicted in **Figure 2**. These representative MMPs were similar among machine learning methods.

1. Overall performance

* BA and MCC values of each ML method with randomly separated data set are shown in boxplot in **Figure 1**
* SVM, RF, FCNN\_sep showed better performance followed by 1NN and 5NN
* Both MPNN model did not show high performance
  + Classical ML methods are enough to build AC prediction model
  + Modern GCN methods are not required in AC prediction.
* Neural networks with multiple input showed better performance than those with concatenated input
  + This is consistent with MMP kernel in SVM which also calculates similarity of core and substituents separately.
* Exemplary MMPs which were predicted accurately with high/low probability are provided in **Figure 2**
  + MMPs with high/low probability are similar among ML methods

1. Effect of training set size

* Swarm plot of MCC colorized along their training data set size is shown in **Figure 3**
* Most of activity classes with large training set are distributed in relatively high area
* However, many activity classes with small set shows high MCC
  + - Using large data set is helpful to build a predictive model but not a critical factor
    - AC prediction is an activity class-specific task

1. Data leakage phenomena

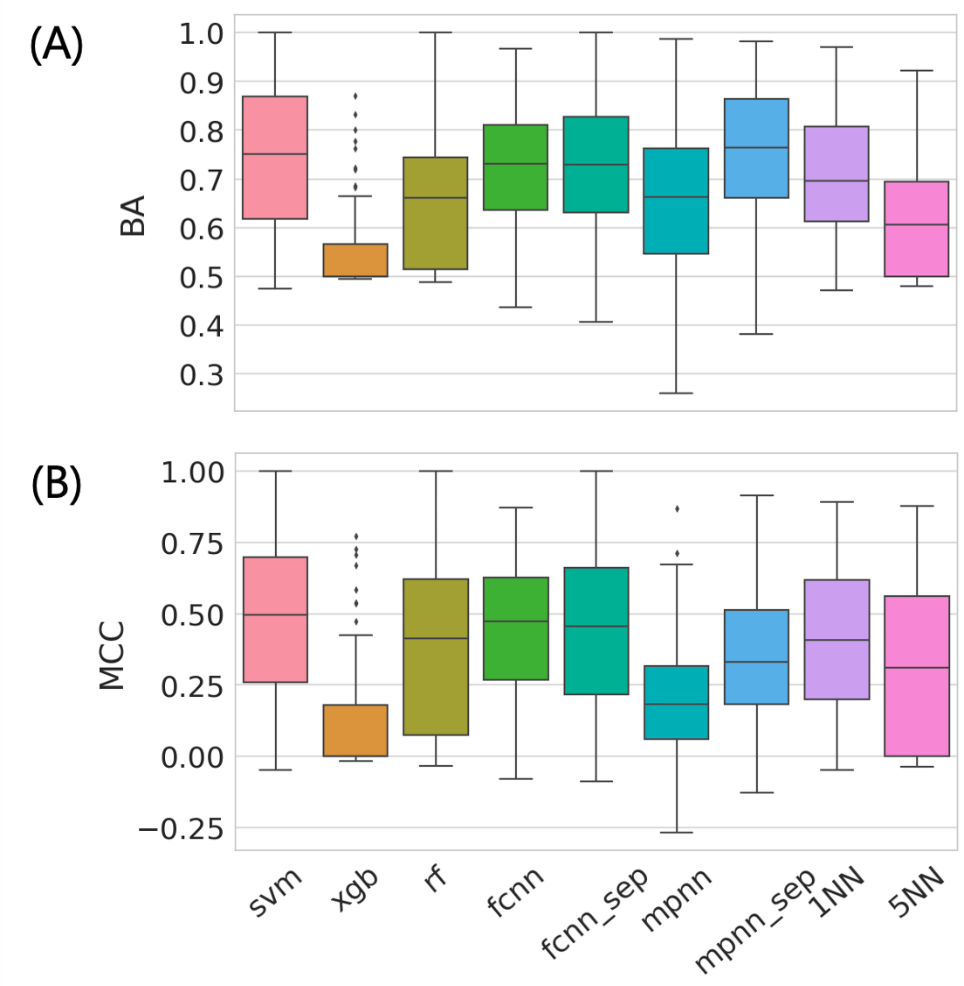
* Boxplots of BA and MCC with two data split method are shown in **Figure 4**
* SVM, both FCNN, MPNN\_sep, 1NN and 5NN showed slightly better than the other when data leakage was excluded
  + Simply evaluating similarity with MMPkernel worked well
* When data leakage was excluded, BA and MCC values decreased regardless of ML methods
  + Sharing same compound in training and test set strongly affect to the performance in AC prediction
  + Data leakage phenomena should be considered in AC prediction

1. Class balance of dataset

* BA, MCC, recall, and precision values of SVM and MPNN\_sep with/without adjusting data balance were provided in **Figure 5** and **Figure 6**, respectively
* SVM and MPNN\_sep were selected to compare the results between classic and modern methods
* Top10 activity classes in the number of MMP-cliff were used for this comparison
* For SVM, recall was improved and precision decreased for both data split strategy when data set was balanced
* For MPNN, recall was improved for data leakage excluded set and precision decreased for data leakage included set
  + Data set balance should be considered based on the user’s project
  + Data set should be balanced if one wants to find MMP-cliff in the given data set as many as possible (recall is prior)
  + All samples should be used if one needs to get reliable MMP-cliffs from prediction (precision is prior)

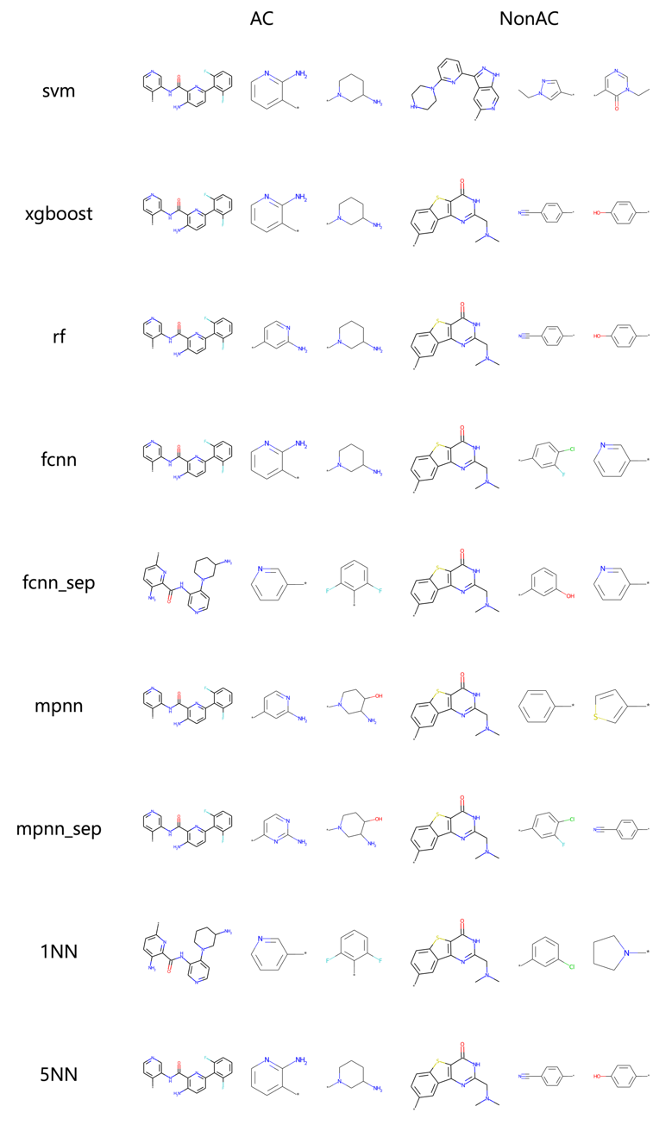
1. Conclusion

* AC prediction is predictive especially with SVM and MMPkernel
* Using large training set is helpful to build predictive model but AC prediction is basically activity class-dependent task
* Data leakage phenomena affect to the model performance so this should be considered during model validation
* Data set balance should be considered along user’s usages of AC prediction



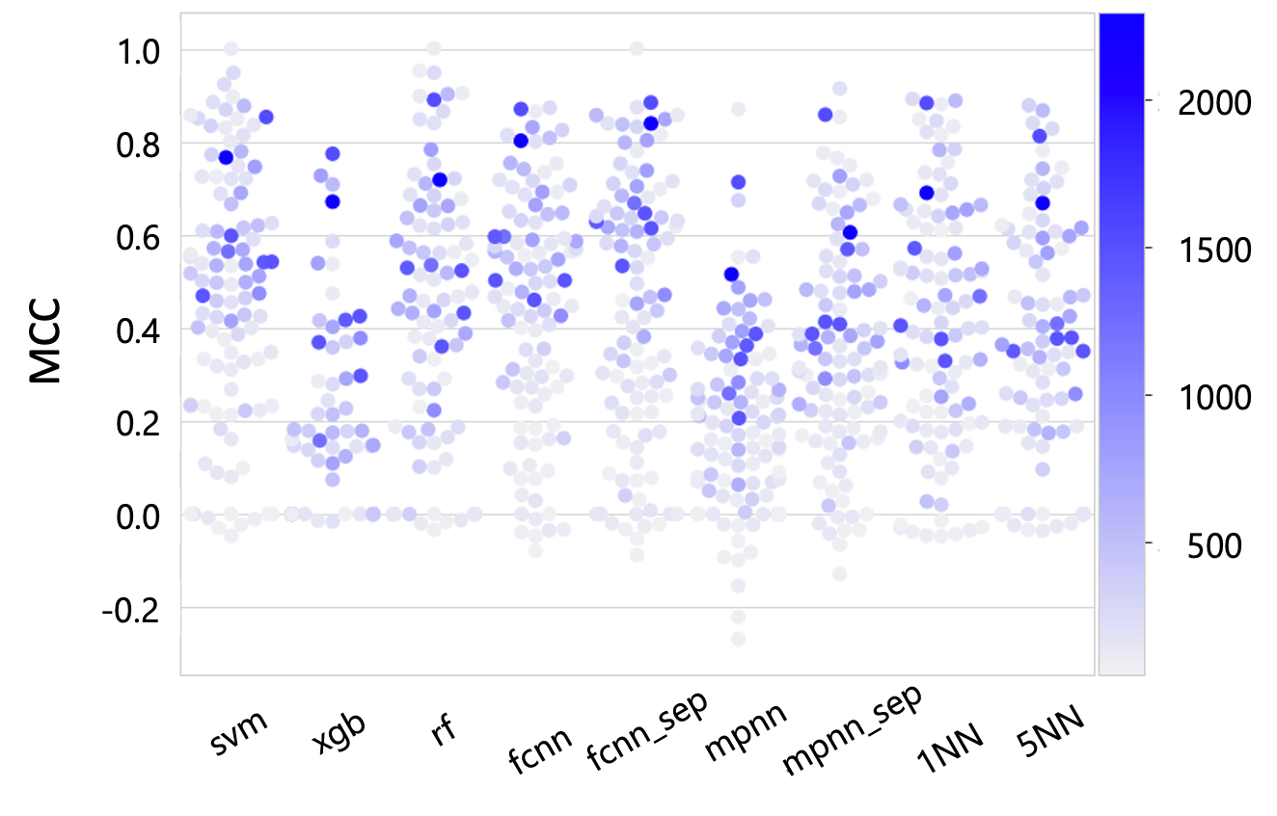
**Figure 1. Global prediction accuracy**

The performance distributions for nine machine learning approach using 100 activity classes are reported in bar plot. The performances were evaluated with (A) balanced accuracy (BA) and (B) Matthew’s correlation coefficient (MCC). Each model was built with data-leakage possibly included (randomly separated) training set.

****

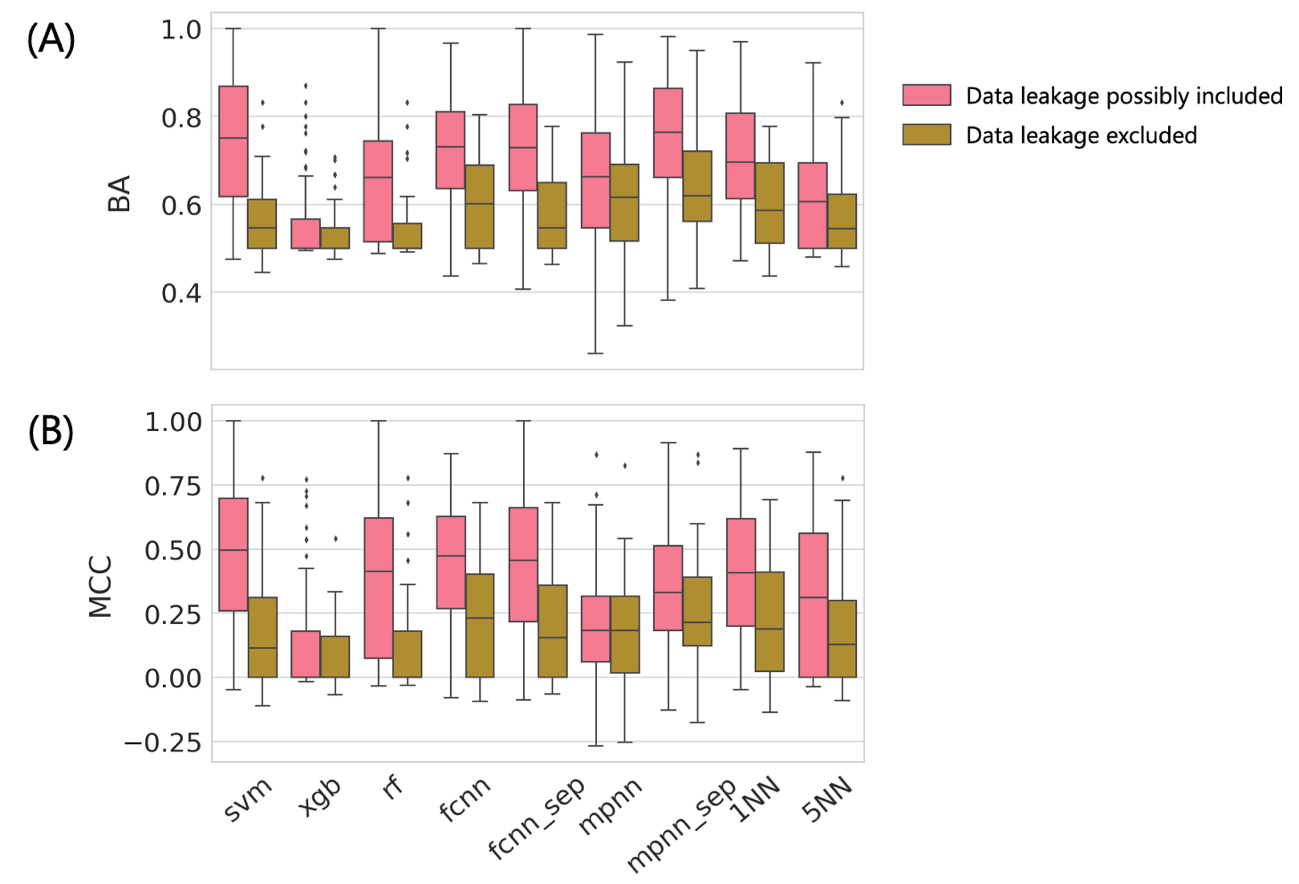
**Figure 2. Exemplary MMPs.**

Exemplary MMPs which were accurately predicted as AC or nonAC by each machine learning approach are depicted. These MMPs are composed of compounds active against ChEMBL4523.



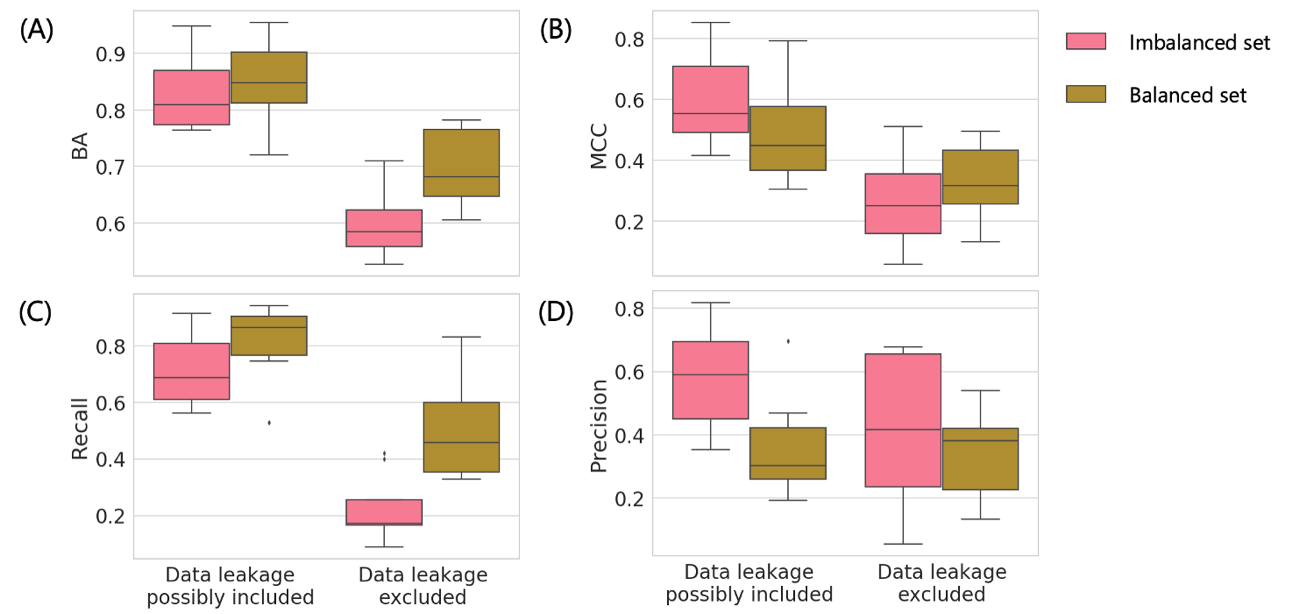
**Figure 3. Effect of training set size**

Swarm plot of MCC values for nine machine learning approach using 100 activity class are colored with blue based on the number of MMPs in the training set. The darker the color is, the more MMPs is in the training set.



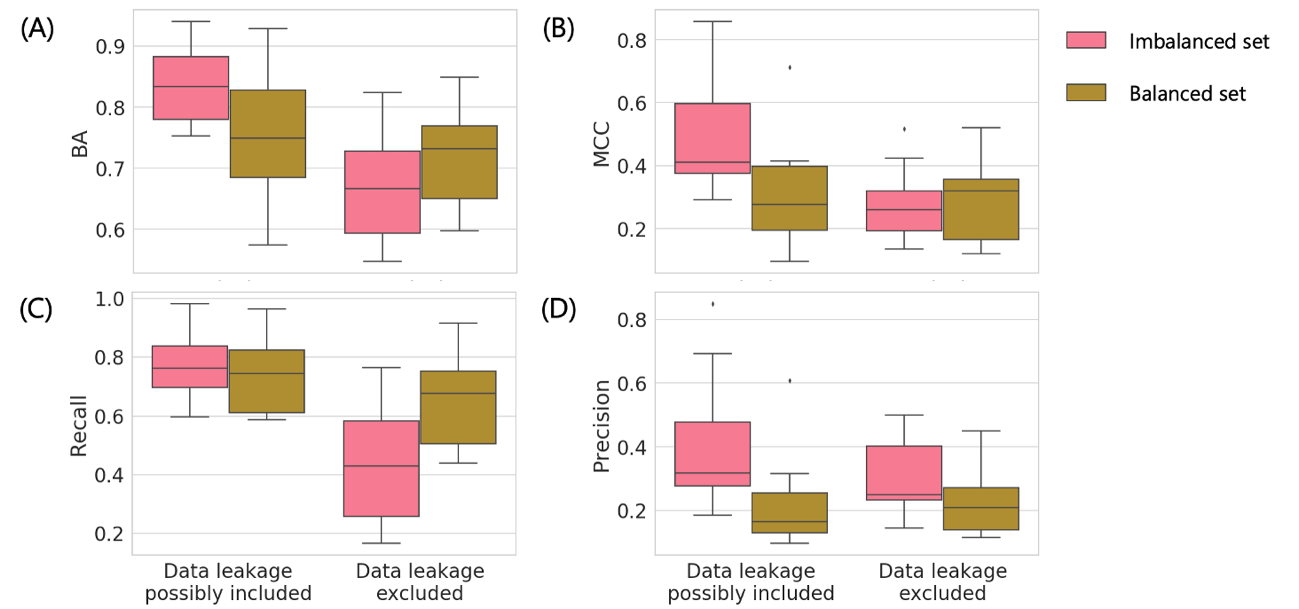
**Figure 4. Comparison of data leakage**

The performance distributions with two distinct data split approach are reported in bar plot. The red bars indicate the performances with data leakage possibly included set and brawn data leakage excluded. The performances were evaluated with (A) balanced accuracy (BA) and (B) Matthew’s correlation coefficient (MCC). The performance distributions for 42 activity classes which were common in the two data set were compared.



**Figure 5. Effect of data set balance for SVM**

The performances with SVM between balanced and imbalanced set were compared in bar plot. The models were built for top 10 activity classes with AC MMPs. The performances were evaluated with (A) BA, (B) MCC, (C) recall, and (D) precision. Generating balanced set was performed for both types of data set; left column: data leakage possibly included set, right: data leakage excluded set. Balanced set was generated by randomly selecting samples from imbalanced set so that the number of AC and non-AC MMPs were same.



**Figure 6. Effect of data set balance for MPNN**

The performances with SVM between balanced and imbalanced set were compared in bar plot. The models were built for top 10 activity classes with AC MMPs. The performances were evaluated with (A) BA, (B) MCC, (C) recall, and (D) precision. Generating balanced set was performed for both types of data set; left column: data leakage possibly included set, right: data leakage excluded set. Balanced set was generated by randomly selecting samples from imbalanced set so that the number of AC and non-AC MMPs were same.

**References**

1. G. M. Maggiora, *J. Chem. Inf. Model.*, 2006, **46**, 1535−1535.
2. D. Stumpfe, Y. Hu, D. Dimova and J. Bajorath, *J. Med. Chem.* 2014, **57**, 18-28.
3. D. Stumpfe, H. Hu and J. Bajorath, *ACS Omega*, 2019, **4**, 14360-14368.
4. R. Guha, *J. Chem. Inf. Model.*, 2012, **2**, 2181-2191.
5. K. Heikamp, X. Hu, A. Yan and J. Bajorath, *J. Chem. Inf. Model.*, 2012, **52**, 2354-2365.
6. S. Tamura, T. Miyao and K. Funatsu, *Mol. Inform.*, 2020, **39**, 2000103.
7. S. Tamura, S. Jasial, T. Miyao and K. Funatsu, *Molecules*, 2021, **26**, 4916.
8. D. Horvath, G. Marcou, A. Varnek, S. Kayastha, A. de la Vega de León and J. Bajorath, *J. Chem. Inf. Model.*, 2016, **56**, 1631-1640.
9. J. Husby, G. Bottegoni, I. Kufareva, R. Abagyan and A. Cavalli, *J. Chem. Inf. Model.*, 2015, **55**, 1062-1076.
10. J. Iqbal, M. Vogt and J. Bajorath, *Artif. Intell. Life Sci.*, 2021, **1**, 100022.
11. J. Iqbal, M. Vogt and J. Bajorath, *J. Comput. Aided. Mol. Des.*, 2021, **35**, 1157-1164.
12. J. Park, G. Sung, S. Lee, S. Kang and C. Park, *J. Chem. Inf. Model*., 2022, **62**, 2341-2351.
13. J. Jiménez-Luna, M. Skalic, N. Weskamp, *J. Chem. Inf. Model*., 2022, **62**, 274-283.
14. D. van Tilborg, A. Alenicheva and F. Grisoni, Exposing the limitations of molecular machine learning with activity cliffs, ChemRxiv preprint, 2022.
15. Hu, H.; Stumpfe, D.; Bajorath, J. Second-Generation Activity Cliffs Identified on the Basis of Target Set-Dependent Potency Difference Criteria. *Future Med Chem* 2019, *11* (5), 379–394. <https://doi.org/10.4155/fmc-2018-0299.>

(1) Mendez, D.; Gaulton, A.; Bento, A. P.; Chambers, J.; De Veij, M.; Félix, E.; Magariños, M. P.; Mosquera, J. F.; Mutowo, P.; Nowotka, M.; Gordillo-Marañón, M.; Hunter, F.; Junco, L.; Mugumbate, G.; Rodriguez-Lopez, M.; Atkinson, F.; Bosc, N.; Radoux, C. J.; Segura-Cabrera, A.; Hersey, A.; Leach, A. R. ChEMBL: Towards Direct Deposition of Bioassay Data. *Nucleic Acids Res* 2018, *47* (D1), gky1075-. <https://doi.org/10.1093/nar/gky1075.>

(2) Hu, X.; Hu, Y.; Vogt, M.; Stumpfe, D.; Bajorath, J. MMP-Cliffs: Systematic Identification of Activity Cliffs on the Basis of Matched Molecular Pairs. *J Chem Inf Model* 2012, *52* (5), 1138–1145. <https://doi.org/10.1021/ci3001138.>

(3) Hussain, J.; Rea, C. Computationally Efficient Algorithm to Identify Matched Molecular Pairs (MMPs) in Large Data Sets. *J Chem Inf Model* 2010, *50* (3), 339–348. <https://doi.org/10.1021/ci900450m.>

(4) Wawer, M.; Bajorath, J. Local Structural Changes, Global Data Views: Graphical Substructure−Activity Relationship Trailing. *J Med Chem* 2011, *54* (8), 2944–2951. <https://doi.org/10.1021/jm200026b.>

(5) Stumpfe, D.; Bajorath, J. Exploring Activity Cliffs in Medicinal Chemistry. *J Med Chem* 2012, *55* (7), 2932–2942. <https://doi.org/10.1021/jm201706b.>

(6) Hu, H.; Stumpfe, D.; Bajorath, J. Second-Generation Activity Cliffs Identified on the Basis of Target Set-Dependent Potency Difference Criteria. *Future Med Chem* 2019, *11* (5), 379–394. <https://doi.org/10.4155/fmc-2018-0299.>

(7) Park, J.; Sung, G.; Lee, S.; Kang, S.; Park, C. ACGCN: Graph Convolutional Networks for Activity Cliff Prediction between Matched Molecular Pairs. *J Chem Inf Model* 2022. <https://doi.org/10.1021/acs.jcim.2c00327.>

(8) Horvath, D.; Marcou, G.; Varnek, A.; Kayastha, S.; León, A. de la V. de; Bajorath, J. Prediction of Activity Cliffs Using Condensed Graphs of Reaction Representations, Descriptor Recombination, Support Vector Machine Classification, and Support Vector Regression. *J Chem Inf Model* 2016, *56* (9), 1631–1640. <https://doi.org/10.1021/acs.jcim.6b00359.>

(9) Heikamp, K.; Hu, X.; Yan, A.; Bajorath, J. Prediction of Activity Cliffs Using Support Vector Machines. *J Chem Inf Model* 2012, *52* (9), 2354–2365. <https://doi.org/10.1021/ci300306a.>

(10) Rogers, D.; Hahn, M. Extended-Connectivity Fingerprints. *J Chem Inf Model* 2010, *50* (5), 742–754. <https://doi.org/10.1021/ci100050t.>

(11) *OEChem TK*; OpenEye Scientific Software Inc, Santa Fe.

(12) Luca, A. de; Horvath, D.; Marcou, G.; Solov’ev, V.; Varnek, A. Mining Chemical Reactions Using Neighborhood Behavior and Condensed Graphs of Reactions Approaches. *J Chem Inf Model* 2012, *52* (9), 2325–2338. <https://doi.org/10.1021/ci300149n.>

(13) *RDKit: Cheminformatics and Machine Learning Software.* <http://www.rdkit.org/> (accessed 2022-10-03).

(14) Pedregosa, F.; Varoquaux, G.; Gramfort, A.; Michel, V.; Thirion, B.; Grisel, O.; Blondel, M.; Müller, A.; Nothman, J.; Louppe, G.; Prettenhofer, P.; Weiss, R.; Dubourg, V.; Vanderplas, J.; Passos, A.; Cournapeau, D.; Brucher, M.; Perrot, M.; Duchesnay, É. Scikit-Learn: Machine Learning in Python. *J Mach Learn Res* 2011, *12* (85), 2825–2830.

(15) Paszke, A.; Gross, S.; Massa, F.; Lerer, A.; Bradbury, J.; Chanan, G.; Killeen, T.; Lin, Z.; Gimelshein, N.; Antiga, L.; Desmaison, A.; Kopf, A.; Yang, E.; DeVito, Z.; Raison, M.; Tejani, A.; Chilamkurthy, S.; Steiner, B.; Fang, L.; Bai, J.; Chintala, S. PyTorch: An Imperative Style, High-Performance Deep Learning Library. *Advances in Neural Information Processing Systems*; Vancouver, Canada, 2019; Vol. 32.

(16) Akiba, T.; Sano, S.; Yanase, T.; Ohta, T.; Koyama, M. Optuna: A Next-Generation Hyperparameter Optimization Framework; The 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining; Anchorage, AK, USA, 2019; pp 2623–2631. <https://doi.org/10.1145/3292500.3330701.>

(17) Vapnik, V. N. The Nature of Statistical Learning Theory. 2000. <https://doi.org/10.1007/978-1-4757-3264-1.>

(18) Ralaivola, L.; Swamidass, S. J.; Saigo, H.; Baldi, P. Graph Kernels for Chemical Informatics. *Neural Networks* 2005, *18* (8), 1093–1110. <https://doi.org/10.1016/j.neunet.2005.07.009.>

(19) Breiman, L. Random Forests. *Mach Learn* 2001, *45* (1), 5–32. <https://doi.org/10.1023/a:1010933404324.>

(20) Friedman, J. H. Greedy Function Approximation: A Gradient Boosting Machine. *Ann Statistics* 2001, *29* (5). <https://doi.org/10.1214/aos/1013203451.>

(21) Vinod, N.; E, H. G. Rectified Linear Units Improve Restricted Boltzmann Machines. *Rectified Linear Units Improve Restricted Boltzmann Machines*; 2010; pp 807–814.

(22) Kingma, D. P.; Ba, J. Adam: A Method for Stochastic Optimization. *Arxiv* 2014. <https://doi.org/10.48550/arxiv.1412.6980.>

(23) Maeda, I.; Sato, A.; Tamura, S.; Miyao, T. Ligand-Based Approaches to Activity Prediction for the Early Stage of Structure–Activity–Relationship Progression. *J Comput Aid Mol Des* 2022, *36* (3), 237–252. <https://doi.org/10.1007/s10822-022-00449-2.>

(24) Tang, B.; Kramer, S. T.; Fang, M.; Qiu, Y.; Wu, Z.; Xu, D. A Self-Attention Based Message Passing Neural Network for Predicting Molecular Lipophilicity and Aqueous Solubility. *J Cheminformatics* 2020, *12* (1), 15. <https://doi.org/10.1186/s13321-020-0414-z.>

(25) Brodersen, K. H.; Ong, C. S.; Stephan, K. E.; Buhmann, J. M. The Balanced Accuracy and Its Posterior Distribution; 2010 20th International Conference on Pattern Recognition; 2010; pp 3121–3124. <https://doi.org/10.1109/icpr.2010.764.>

(26) Matthews, B. W. Comparison of the Predicted and Observed Secondary Structure of T4 Phage Lysozyme. *Biochimica Et Biophysica Acta Bba - Protein Struct* 1975, *405* (2), 442–451. <https://doi.org/10.1016/0005-2795(75)90109-9.>