Ann Math Artif Intell (2017) 81:105–123
DOI 10.1007/s10472-017-9556-8

Conformal prediction of biological activity of chemical compounds

Paolo Toccaceli[1] · **Ilia Nouretdinov**[1] · **Alexander Gammerman**[1]

Published online: 16 June 2017

Abstract The paper presents an application of Conformal Predictors to a chemoinformatics problem of predicting the biological activities of chemical compounds. The paper addresses some specific challenges in this domain: a large number of compounds (training examples), high-dimensionality of feature space, sparseness and a strong class imbalance. A variant of conformal predictors called Inductive Mondrian Conformal Predictor is applied to deal with these challenges. Results are presented for several non-conformity measures extracted from underlying algorithms and different kernels. A number of performance measures are used in order to demonstrate the flexibility of Inductive Mondrian Conformal Predictors in dealing with such a complex set of data. This approach allowed us to identify the most likely active compounds for a given biological target and present them in a ranking order.

Keywords Conformal prediction · Confidence estimation · Chemoinformatics · Non-conformity measure

Mathematics Subject Classification (2010) 68T05

✉ Paolo Toccaceli
Paolo.Toccaceli@rhul.ac.uk

Ilia Nouretdinov
ilia@cs.rhul.ac.uk

Alexander Gammerman
alex@cs.rhul.ac.uk

[1] Royal Holloway, University of London, Egham, UK

1 Introduction

Compound Activity Prediction is one of the key research areas of Chemoinformatics. It is of critical interest for the pharmaceutical industry, as it promises to cut down the costs of drug discovery by reducing the number of lab tests needed to identify a bioactive compound among a vast set of candidates (hit or lead generation [17]). In particular, the approach relevant to this paper is generally known as Quantitative Structure-Activity Relationship (QSAR), where the activity of a compound is predicted from its chemical structure. The focus is on providing a set of potentially active compounds that is significantly "enriched" in terms of prevalence of bioactive compounds compared to a purely random sample of the compounds under consideration. The paper is an extension of our work presented in [20].

While it is true that this objective in itself could be helped with the classical machine learning techniques that usually provide a bare prediction, the hedged predictions made by Conformal Predictors (CP) provide some additional information that can be used advantageously in a number of respects.

First, CPs will supply the valid measures of confidence in the prediction of bioactivities of the compounds. Second, they can provide prediction and confidence for individual compounds. Third, they can allow the ranking of compounds to optimize the experimental testing of given samples. Finally, the user can control the number of errors and other performance measures like precision and recall by setting up a required level of confidence in the prediction.

However, the realization of these benefits when applying CPs to QSAR data is challenging because of the size ($\gg$100k examples) and the imbalance of the data sets ($\approx$1% active compounds) emerging from High-Throughput Screening. These two challenges are met with the use of Inductive and Mondrian CP respectively.

2 Machine learning background

2.1 Conformal predictors

Conformal Predictors [9, 15] revolve around the notion of Conformity (or rather of Non-Conformity). Intuitively, one way of viewing the problem of classification is to assign a label $\hat{y}$ to a new object x so that the example $(x, \hat{y})$ does not look out of place among the training examples $(x_1, y_1), (x_2, y_2), \ldots, (x_\ell, y_\ell)$. To find how "strange" the new example $(x, \hat{y})$ is in comparison with the training set, we use the Non-Conformity Measure (NCM). The advantage of approaching classification in this way is that this leads to a novel way to quantify the uncertainty of the prediction, under some rather general hypotheses.

A Non-Conformity Measure can be extracted from any machine learning algorithm, although there is no universal method to choose it. Note that we are not necessarily interested in the actual classification resulting from such "underlying" machine learning algorithm. What we are really interested in is an indication of how "unusual" an example appears, given a training set.

We assume here that our training data are IID (*independent and identically distributed* data).[1] Armed with an NCM, it is possible to compute for any example (x, y) a *p-value* that reflects how well the new example from the test set fits (or *conforms*) with the IID

[1] In fact, the IID assumption can be replaced by a weaker assumption of *exchangeability*.

assumption of the training data. A more accurate and formal statement is: for a chosen $\epsilon \in [0, 1]$ it is possible to compute p-values for test objects so that they are (in the long run) smaller than ϵ with probability at most ϵ.

The idea is then to compute for a test object a p-value of every possible choice of the label.

Once the p-values are computed, they can be put to use in one of the following ways:

- Given a significance level, ϵ, a *region predictor* outputs for each test object the set of labels (i.e., a region in the label space) such that the actual label is not in the set no more than a fraction ϵ of the times. If the prediction set consists of more than one label, the prediction is called *uncertain*, whereas if there are no labels in the prediction set, the prediction is *empty*.
- Alternatively, a *forced* prediction (chosen by the largest p-value) is given, alongside with its *credibility* (the largest p-value) and *confidence* (the complement to 1 of the second largest p-value).

2.2 Inductive mondrian conformal predictors

In order to apply conformal predictors to both big and imbalanced datasets, we combine two variants of conformal predictors from [9, 15]: Inductive (to reduce computational complexity) and Mondrian (in its so called *label-conditional* version to deal with imbalanced data sets) Conformal Predictors.

To combine the Mondrian Conformal Prediction with that of Inductive Conformal Prediction, we have to revise the definition of p-value for the Mondrian case so that it incorporates the changes brought about by splitting the training set and evaluating the α_i only in the calibration set, where α_i is a measure of strangeness or NCM measure. It is customary to denote the test object with $x_{\ell+1}$ where ℓ is the size of the training set and to split the training set at index h so that examples with index $i \leq h$ constitute the proper training set and examples with index $i > h$ (and $i \leq \ell$) constitute the calibration set. The proper training set is used to train the underlying Machine Learning algorithm, which is then used to calculate the NCM $\alpha_{h+1}, \ldots, \alpha_\ell$ on the examples in the calibration set.

The p-values for a hypothesis $y_{\ell+1} = \overline{y}$ about the label of $x_{\ell+1}$ are defined as

$$p(\overline{y}) = \frac{|\{i = h+1, \ldots, \ell+1 : y_i = \overline{y}, \alpha_i \geq \alpha_{\ell+1}\}|}{|\{i = h+1, \ldots, \ell+1 : y_i = \overline{y}\}|}$$

In words, the formula above computes the p-value for a label assignment $\overline{y}$ to the test object $x_{\ell+1}$ as the fraction of examples with label $\overline{y}$ in the calibration set that have the same or larger Non-Conformity Measure as the hypothetical test example $(x_{\ell+1}, \overline{y})$.

This will allow us to use NCM based on such methods as Cascade SVM (described in Section 3.1), including a stage of splitting big data into parts.

3 Application to compound activity prediction

To evaluate the performance of CP for Compound Activity Prediction in a realistic scenario, we sourced the data sets from a public-domain repository of High Throughput assays, PubChem BioAssay [22].

The data sets on PubChem identify a compound with its CID (a unique compound identifier that can be used to access the chemical data of the compound in another PubChem

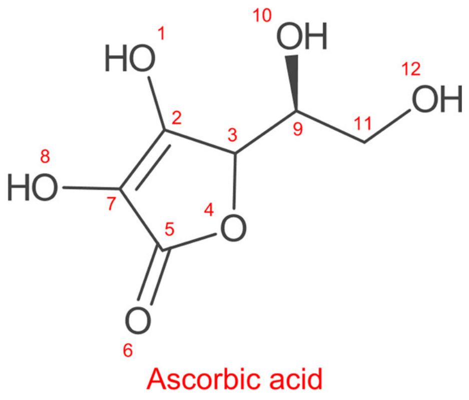

Fig. 1 Chemical structure of *l*-ascorbic acid, commonly known as vitamin C. Consistently with convention in organic chemistry, carbon atoms and hydrogen atoms are not indicated as their presence can be easily inferred (carbon atoms are at every unlabelled vertex, hydrogen atoms are present wherever needed to saturate the valence of an atom). The numbering of the atoms in this example is arbitrary

database) and provide the result of the assay as Active/Inactive as well as providing the actual measurements on which the result was derived, e.g. viability (percentage of cells alive) of the sample after exposure to the compound.

To apply machine learning techniques to this problem, the compounds must be described in terms of a number of numerical attributes. There are several approaches to do this. The approach that was followed in this study is to compute *signature descriptors*[2] [8]. Each signature corresponds a specially constructed directed acyclic labelled subgraph in the molecule graph. Signatures have a height, which corresponds to the number of edges between root and terminal nodes of the directed acyclic subgraphs. In this exercise the signatures had at most height 3. The signature descriptor for a molecule consists of the signatures present in the molecule along with their counts, i.e. the number of times the labelled subgraph of a signature occur in the graph of the molecule. An example of the signature descriptor for ascorbic acid (also known as vitamin C) is provided in Fig. 1, Tables 1 and 2.

To create a data set from a number of compounds, all the signatures in all compounds are first enumerated and the set of all signatures is obtained. Each unique signature corresponds to one attribute, hence one dimension of the data set. To build the matrix of training examples, each signature in the set is attributed (arbitrarily) a column index and each compound a row index. Each cell of the matrix contains the count of the occurrences of the signature corresponding to the column in the compound corresponding to the row. The resulting matrix can be very large but it is also highly sparse, as detailed further on (see Table 4).

3.1 Underlying algorithms

As a first step in the study, we set out to extract relevant non-conformity measures from different underlying algorithms: Support Vector Machines (SVM), Nearest Neighbours, Naïve Bayes. The Non Conformity Measures for each of the three underlying algorithms are listed in Table 3.

There are a number of considerations arising from the application of each of these algorithms to Compound Activity Prediction.

[2]The signature descriptors and other types of descriptors (e.g. circular descriptors) can be computed with the CDK (Chemistry Development Kit) Java package or any of its adaptations such as the RCDK package for the R statistical software.

Table 1 Signatures for ascorbic acid

	0	1	2	3
1	[O]	[O]([C])	[O]([C]([C]=[C]))	[O]([C]([C]([C][O])=[C]([C][O])))
2	[C]	[C]([C]=[C][O])	[C]([C]([C][O])=[C]([C][O])[O])	[C](=[C]([C](=[O][O,0])[O])[C]([C]([C][O])[O,0...
3	[C]	[C]([C][C][O])	[C]([C]([C][O])[C](=[C][O])[O]([C]))	[C]([C]([C]([O])[O])[C](=[C]([C,0][O])[O])[O](...
4	[O]	[O]([C][C])	[O]([C]([C][C])[C]([C]=[O]))	[O]([C]([C](=[C,0][O])=[O])[C]([C]([C][O])[C,0...
5	[C]	[C]([C][O]=[O])	[C]([C](=[C][O])=[O][O]([C]))	[C]([C](=[C]([C,0][O])[O])=[O][O]([C,0]([C])))
6	[O]	[O](=[C])	[O](=[C]([C][O]))	[O](=[C]([C](=[C][O])[O]([C])))
7	[C]	[C]([C]=[C][O])	[C](=[C]([C][O])[C]([O]=[O])[O])	[C]([C](=[O][O]([C,0]))=[C]([C,0]([C])[O])[O])
8	[O]	[O]([C])	[O]([C]([C]=[C]))	[O]([C](=[C]([C][O])[C]([O]=[O])))
9	[C]	[C]([C][C][O])	[C]([C]([O])[C]([C][O])[O])	[C]([C]([O])[C]([C](=[C][O])[O]([C]))[O])
10	[O]	[O]([C])	[O]([C]([C][C]))	[O]([C]([C]([O])[C]([C][O])))
11	[C]	[C]([C][O])	[C]([C]([C][O])[O])	[C]([C]([C]([C][O])[O])[O])
12	[O]	[O]([C])	[O]([C]([C]))	[O]([C]([C]([C][O])))

For every “heavy” atom (i.e. every non-hydrogen atom), a labelled Directed Acyclic Graph (DAG) of given depth is computed and a string-based representation is provided. For instance, the string at row 7 and column 1 represents the DAG of depth 1 from atom 7, i.e. a carbon atom with a bond to a carbon atom, a double bond to another carbon atom and a bond to an oxygen atom

Table 2 Signature descriptor for ascorbic acid

Counts	Signature
6	[C]
6	[O]
4	[O]([C])
2	[C]([C]=[C][O])
2	[C]([C][C][O])
2	[O]([C]([C]=[C]))
1	[C](=[C]([C](=[O][O,0])[O])[C]([C]([C][O])[O,0])[O])
1	[C](=[C]([C][O])[C]([O]=[O])[O])
1	[C]([C](=[C]([C,0][O])[O])=[O][O]([C,0]([C])))
1	[C]([C](=[C][O])=[O][O]([C]))
1	[C]([C](=[O][O]([C,0]))=[C]([C,0]([C])[O])[O])
1	[C]([C]([C]([C][O])[O])[O])
1	[C]([C]([C]([O])[O])[C](=[C]([C,0][O])[O])[O]([C,0](=[O])))
1	[C]([C]([C][O])=[C]([C][O])[O])
1	[C]([C]([C][O])[C](=[C][O])[O]([C]))
1	[C]([C]([C][O])[O])
1	[C]([C]([O])[C]([C](=[C][O])[O]([C]))[O])
1	[C]([C]([O])[C]([C][O])[O])
1	[C]([C][O])
1	[C]([C][O]=[O])
1	[O](=[C]([C](=[C][O])[O]([C])))
1	[O](=[C]([C][O]))
1	[O](=[C])
1	[O]([C](=[C]([C][O])[C]([O]=[O])))
1	[O]([C]([C](=[C,0][O])=[O])[C]([C]([C][O])[C,0]([O])))
1	[O]([C]([C]([C][O])))
1	[O]([C]([C]([C][O])=[C]([C][O])))
1	[O]([C]([C]([O])[C]([C][O])))
1	[O]([C]([C]))
1	[O]([C]([C][C]))
1	[O]([C]([C][C])[C]([C]=[O]))
1	[O]([C][C])

Each signature is used as a feature (a dimension); the number of occurrences is the value of the feature

SVM The usage of SVM in this domain poses a number of challenges. First of all, the number of training examples was large enough to create a problem for our computational resources. The scaling of SVM to large data sets is indeed an active research area [2, 7, 18, 19]. We turned our attention to a simple approach proposed by V.Vapnik et al. in [11], called Cascade SVM.

The sizes of the training sets considered here are too large to be handled comfortably by generally available SVM implementations, such as `libsvm` [6]. The approach we follow could be construed as a form of *training set editing*. Vapnik proved formally that it is possible to decompose the training into an n-ary tree of SVM trainings. The first layer of SVMs

Table 3 The non conformity measures for the three underlying algorithms

Underlying	Non conformity measure α_i	Comment
Support Vector Machine (SVM)	$-y_i d(x_i)$	(signed) distance between the example and the separating hyperplane
k Nearest Neighbours (kNN)	$\frac{\sum_{j \neq i: y_j = y_i}^{(k)} d(x_j, x_i)}{\sum_{j \neq i: y_j \neq y_i}^{(k)} d(x_j, x_i)}$	Here the summation is on the k smallest values of the distance between the example and its neighbours from same/other class
Naïve Bayes	$-\log\ p(y_i = c \vert x_i)$	p is the posterior probability of the example's class estimated by Naïve Bayes

is trained on training sets obtained as a partition of the overall training set. Each SVM in the first layer outputs its set of support vectors (SVs) which is generally smaller than the training set. In the second layer, each SVM takes as training set the merging of n of the SVs sets found in the first layer. Each layer requires fewer SVMs. The process is repeated until a layer requires only one SVM. The set of SVs emerging from the last layer is not necessarily the same that would be obtained by training on the whole set (but it is often a good approximation). If one wants to obtain that set, the whole training tree should be executed again, but this time the SVs obtained at the last layer would be merged into each of the initial training blocks. A new set of SVs would then be obtained at the end of the tree of SVMs. If this new set is the same as the one in the previous iteration, this is the desired set. If not, the process is repeated once more. In [11] it was proved that the process converges and that it converges to the same set of SVs that one would obtain by training on the whole training set in one go.

To give an intuitive justification, the fundamental observation is that the SVM decision function is entirely defined just by the Support Vectors. It is as if these examples contained all the information necessary for the classification. Moreover, if we had a training set composed only of the SVs, we would have obtained the same decision function. So, one might as well remove the non-SVs altogether from the training set.

In experiments discussed here, we followed a simplified approach. Instead of a tree of SVMs, we opted for a linear arrangement as shown in Fig. 2.

While we have no theoretical support for this semi-online variant of the Cascade SVM, the method appears to work satisfactorily in practice on the data sets we used.

The class imbalance was addressed with the use of per-class weighting of the C hyper-parameter, which results in a different penalization of the margin violations. The per-class weight was set inversely proportional to the class representation in the training set.

Another feature of SVM is the choice of an appropriate kernel. While we appreciated the computational advantages of linear SVM, we also believed that it was not necessarily the best choice for the specific problem. It can easily be observed that the nature of the representation of the training objects (as discrete features) warranted approaches similar to those used in Information Retrieval, where objects are described in terms of occurrences of patterns (bags of words). The topic of similarity searching in chemistry is an active one and there are many alternative proposals (see [1]). We used as a kernel a function called Tanimoto similarity[3] defined as:

$$T(A, B) = \frac{\sum_{i=1}^{d} \min(a_i, b_i)}{\sum_{i=1}^{d} (a_i + b_i) - \sum_{i=1}^{d} \min(a_i, b_i)}$$

[3]See [10] for a proof that Tanimoto Similarity is a kernel.

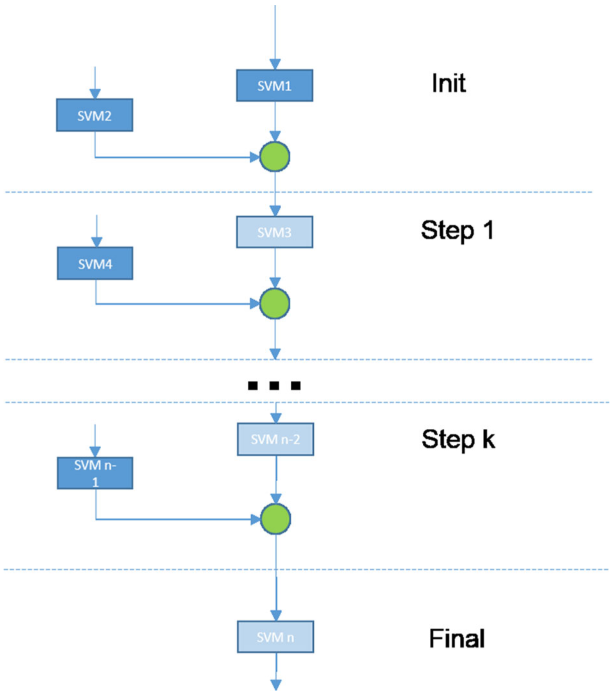

Fig. 2 Linear cascade SVM At each step, the set of support vectors from the previous stage is merged with a block of training examples from the partition of the original training set. This is used as training set for an SVM, whose SVs are then fed to the next stage

where $A = (a_1, \ldots, a_d)$, $B = (b_1, \ldots, b_d)$ are two objects, each described by a vector of d integers. In our context, the integers are counts of the number of times a specific molecular sub-graph occurs in the given molecule. See Table 2 where the sub-graphs are represented with strings with a special syntax.

The Tanimoto similarity extends the well-known Jaccard coefficient in the sense that whereas the Jaccard coefficient considers only presence or absence of a pattern, the Tanimoto similarity takes into account the counts of the occurrences.

To explore further the benefits of non-linear kernels, we also tried out a kernel consisting of the composition the Tanimoto similarity defined above with the Gaussian Radial Basis Function (RBF), which can be expressed as.

$$TG(A, B) = e^{-\frac{|T(A,A)+T(B,B)-2T(A,B)|}{\gamma}}$$

where γ is a positive constant.

Nearest neighbours We chose Nearest Neighbours because of its good performance in a wide variety of domains. In principle, the performance of Nearest Neighbours could be severely affected by the high-dimensionality of the training set (Table 4 shows how in one of the data sets used in this study the number of attributes exceeds by $\approx 20\%$ the number of examples), but in our experiments the "curse of dimensionality" did not manifest itself. In this study the parameter of Nearest Neighbours that is the number of neighbours is set to $k = 3$.

Table 4 Characteristics of the AID827 data set

Total number of examples	138,287	
Number of features	165,786	High dimensionality
Number of non-zero entries	7,711,571	
Density of the data set	0.034%	High sparsity
Active compounds	1,658	High imbalance (1.2%)
Inactive compounds	136,629	
Unique set of signatures	137,901	Low degeneracy[a]

[a]As an element of clarification, *degeneracy* here refers to the undesirable occurrence of the same representation in terms of features for different molecules. Low degeneracy is therefore a good property (whereas the other comments in the table highlight challenges)

Naïve Bayes Naïve Bayes and more specifically Multinomial Naïve Bayes are widely regarded as effective classifiers when features are discrete (for instance, in text classification), despite their relative simplicity. This made Multinomial Naïve Bayes a natural choice for the problem at issue here.

In addition, Naïve Bayes has a potential of providing some guidance for feature selection, via the computed posterior probabilities. This is of particular interest in the domain of Compound Activity Prediction, as it may provide insight as to the molecular structures that are associated with Activity in a given assay. This knowledge could steer further testing in the direction of a class of compounds with higher probability of Activity.

3.2 Tools and computational resources

The choice of the tools for these experiments was influenced primarily by the exploratory nature of this work. For this reason, programming languages and development environments were chosen for their ability to support interactivity and rapid prototyping, rather than optimal CPU utilization and memory efficiency.

The language adopted was Python 3.4 and the majority of programming was done using Jupyter Notebooks [5] with the IPython kernel [4]. The Jupyter Notebooks provide a convenient environment for experimental scientific programming on local as well as on remote systems. Inspired by Mathematica^{TM} Notebooks, they are structured as a sequence of cells, each of which can contain code, text, HTML or other graphics. Code cell can be executed in an underlying Python interpreter (other languages, such as R and Julia are also supported) and its output (including charts) is shown underneath the cell itself. Text cells support a widely-used simple formatting language called Markdown enhanced with the ability to render $\TeX$ formulas. The Jupyter environment is based on a client-server model, in which the client is a browser application which handles all the interaction with the user and the server, located locally or remotely, performs the actual computations. The overall format turned out to be very effective for developing methods, performing analyses on-the-fly, running heavy loads taking advantage of remote computing facilities, and finally capturing results (in a way that facilitates their future reproducibility).

Several third-party libraries were used. Numerical processing and data management were performed with the help of `numpy/scipy` and `pandas`. SVM and other machine learning algorithms were provided by the `scikit-learn` [13] package. Sections of Python code (e.g. the Tanimoto similarity) that were identified through profiling as critical for performance were re-implemented in `Cython`, a subset/dialect of Python that allows code to be compiled (via an intermediate C representation) rather than being interpreted. The computations were run initially on a local server (8 cores with 32GB of RAM, running OpenSuSE) and in later stages on a supercomputer, the IT4I Salomon cluster located in Ostrava, Czech Republic. The Salomon cluster is based on the SGI ICE X system and comprises 1008 computational nodes (plus a number of login nodes), each with 24 cores (2 12-core Intel Xeon E5-2680v3 2.5GHz processors) and 128GB RAM, connected via high-speed 7D Enhanced hypercube InfiniBand FDR and Ethernet networks. It currently ranks at #48 in the top500.org list of supercomputers and at #14 in Europe.[4]

Parallelization and computation distribution relied on the `ipyparallel` [3] package, which is a high-level framework for the coordination of remote execution of Python functions on a generic collection of nodes (cores or separate servers). While `ipyparallel` may not be highly optimized, it aims at providing a convenient environment for distributed computing well integrated with IPython and Jupyter and has a learning curve that is not as steep as that of the alternative frameworks common in High Performance Computing (OpenMPI, for example). In particular, `ipyparallel`, in addition to allowing the start-up and shut-down of a cluster comprising a controller and a number of engines where the actual processing (each is a separate process running a Python interpreter) is performed via integration with the job scheduling infrastructure present on Salomon (PBS, Portable Batch System), took care of the details such as data serialization/deserialization and transfer, load balancing, job tracking, exception propagation, etc. thereby hiding much of the complexity of parallelization. One key characteristic of `ipyparallel` is that, while it provides primitives for `map()` and `reduce()`, it does not constrain the choice to those two, leaving the implementer free to select the most appropriate parallel programming design patterns for the specific problem (see [23] for a reference on the subject).

In this work, parallelization was exploited to speed up the computation of the Gram matrix or of the decision function for the SVMs or the matrix of distances for kNN. In either case, the overall task was partitioned in smaller chunks that were then assigned to engines (each running on a core on a node), which would then asynchronously return the result. Also, parallelization was used for SVM cross-validation, but at a coarser granularity, i.e. one engine per SVM training with a parameter. Data transfers were minimized by making use of shared memory where possible and appropriate. A key speed-up was achieved by using pre-computed kernels (computed once only) when performing Cross-Validation with respect to the hyperparameter C.

A zip archive with the code and the data files used to obtain the results presented in this paper is available at http://clrc.rhul.ac.uk/people/ptocca/AMAI-2017/20170113-AMAI-Package.zip.

3.3 Results

To assess the relative merits of the different underlying algorithms, we applied Inductive Mondrian Conformal Predictors on data set AID827, whose characteristics are listed in Table 4.

The test was articulated in 20 cycles of training and evaluation. In each cycle, a test set of 10,000 examples was extracted at random. The remaining examples were split randomly into a proper training set of 100,000 examples and a calibration set with the balance of the examples (28,387).

[4]According to https://www.sgi.com/company_info/newsroom/press_releases/2015/september/salomon.html.

During the SVM training, 5-fold stratified cross validation was performed at every stage of the Cascade to select an optimal value for the hyperparameter C. Also, per-class weights were assigned to cater for the high class imbalance in the data, so that a higher penalization was applied to violators in the less represented class.

In Multinomial Naïve Bayes too, cross validation was used to choose an optimal value for the smoothing parameter.

The results are listed in Table 5, which presents the classification arising from the region predictor for $\epsilon = 0.01$. The numbers are averages over the 20 cycles of training and testing.

Note that a compound is classified as Active (resp. Inactive) if and only if Active (resp. Inactive) is the only label in the prediction set. When both labels are in the prediction, the prediction is considered Uncertain.

It has to be noted at this stage that there does not seem to be an established consensus on what the best performance criteria are in the domain of Compound Activity Prediction (see for instance [12]), although *Precision* (fraction of actual Actives among compounds predicted as Active) and *Recall* (fraction of all the Active compounds that are among those predicted as Active) seem to be generally relevant. In addition, it is worth pointing out that these (and many other) criteria of performance should be considered as generalisations of classical performance criteria since they include dependence of the results on the required confidence level.

At the shown significance level of $\epsilon = 0.01$, 34% of the compounds predicted as Active by Inductive Mondrian Conformal Prediction using Tanimoto composed with Gaussian RBF were actually Active compared to a prevalence of Actives in the data set of just 1.2%. At the same time, the Recall was $\approx$ 41% (ratio of Actives in the prediction to total Actives in the test set).

We selected Cascade SVM with Tanimoto+RBF as the most promising underlying algorithm on the basis of the combination of its high Recall (for Actives) and high Precision (for Actives), assuming that the intended application is indeed to output a selection of compounds that has a high prevalence of Active compounds.

Note that in Table 5 the values similar to ones of confusion matrix are calculated only for certain predictions. In this representation, the concrete meaning of the property of class-based validity can be clearly illustrated as in Table 6: the two rightmost columns report the

Table 5 Conformal predictors results for AID827 with significance $\epsilon = 0.01$

Underlying	Active pred Active	Inactive pred Active	Inactive pred Inactive	Active pred Inactive	Empty pred	Uncertain
Naïve Bayes	38.20	104.30	183.30	1.10	0	9673.10
3NN	43.95	100.55	361.55	0.80	0	9493.15
Cascade SVM:						
– linear	34.20	99.00	591.85	1.20	0	9273.75
– RBF kernel	47.20	101.80	1126.75	1.80	0	8722.45
– Tanimoto kernel	48.45	97.65	986.85	0.80	0	8866.25
– Tanimoto-RBF kernel	47.65	94.10	1044.90	0.95	0	8812.40

All results are averages over 20 runs, using the same test sets of 10,000 objects across the different underlying algorithms. “Active predicted Active” is the (average) count of actually Active test examples that were predicted Active by Conformal Prediction. Uncertain predictions occur when both labels are output by the region predictor. Empty predictions occur when both labels can be rejected at the chosen significance level. For the specific significance level chosen here, there were never empty predictions

Table 6 Conformal predictors results for AID827 using SVM with Tanimoto+RBF kernel for different significance levelson test sets with 10,000 compounds, out of which 115 were on average Active

Significance	Active pred Active	Inactive pred Active	Inactive pred Inactive	Active pred Inactive	Empty pred Rate	Uncertain Rate	Active Error	Inactive Error
1%	47.65	94.10	1044.90	0.95	0.0	8812.40	0.82%	0.95%
5%	67.20	490.40	3091.75	5.20	0.0	6345.45	4.52%	4.96%
10%	76.15	999.25	4703.75	10.60	0.0	4210.25	9.22%	10.11%
15%	82.10	1484.85	6021.80	17.30	0.0	2393.95	15.04%	15.02%
20%	86.55	1982.25	6928.95	22.80	0.0	979.45	19.83%	20.05%

The results are averages over 20 random splits between test set and training+calibration set. The last two columns in the table demonstrate the *validity* property of conformal predictors. The "Active Error Rate" is the ratio of "Active predicted Inactive" to the total number of Active test examples. The "Inactive Error Rate" is the ratio of "Inactive predicted Active" to the total number of Inactive test examples

prediction error rate for each label, where by prediction error we mean the occurrence of "the actual label not being in the predictions set". When there are no Empty predictions, the Active Error rate is the ratio of the number of "Active predicted Inactive" to the number of Active examples in the test set (which was 115 on average).

Figure 3 shows the test objects according to the base-10 logarithm of their p_{active} and $p_{inactive}$. The dashed lines represent the thresholds for p-value set at 0.01, i.e. the

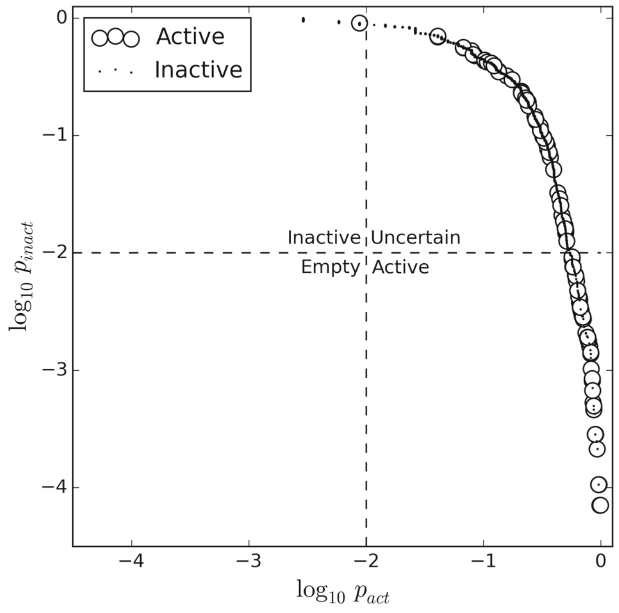

Fig. 3 Test objects plotted by the base-10 log of their their p_{active} and $p_{inactive}$. *Dots* represent test objects that are known to be inactive and *circles* represent those that are known to be active. Note that many test objects are overlapping. Note that some of the examples may have identical $p-$values, so for example 1135 objects predicted as "Inactives" are presented as just 4 points on this plot (those in the upper left quadrant, having $p_{inactive} < 0.01$ and having $p_{active} > 0.01$)

significance value ϵ used in Table 5. The two dashed lines partition the plane in 4 regions, corresponding to the region prediction being Active ($p_{active} > \epsilon$ and $p_{inactive} \leq \epsilon$), Inactive ($p_{active} \leq \epsilon$ and $p_{inactive} > \epsilon$), Empty ($p_{active} \leq \epsilon$ and $p_{inactive} \leq \epsilon$), Uncertain ($p_{active} > \epsilon$ and $p_{inactive} > \epsilon$).

As we indicated in Section 2.1, the alternative is forced prediction with individual confidence and credibility.

It is clear that there are several benefits accruing from using Conformal Predictors. For instance, a high p-value for the Active hypothesis might suggest that Activity cannot be ruled out, but the same compound may exhibit also a high p-value for the Inactive hypothesis, which would actually mean that neither hypothesis could be discounted.

In this specific context it can be argued that the p-values for Active hypothesis are more important. They can be used to rank the test compounds like it was done in [21] for ranking potential interaction. A high p-value for the Active hypothesis might suggest that Activity cannot be ruled out. For example it is possible to output the prediction list of all compounds with p-values above a threshold $\epsilon = 0.01$. A concrete activity which is not yet discovered will be covered by this list with probability 0.99. All the remaining examples are classified as Non-Active with confidence 0.99 or larger.

Special attention should be also paid to *low credibility* examples where both p-values are small. Such examples might appear in the "Empty" quarter of the plot on Fig. 3 if the threshold were changed. For such examples, the label assignment does not conform to the training data. They may be considered as anomalies or examples of compound types not enough represented in the training set. This may suggest that it would be beneficial to the overall performance of the classifier to perform a lab test for those compounds and include the results in training set.

Finally, Conformal Predictors provide the user with the additional degree of freedom of the significance or confidence level. By varying either of those two parameters, a different trade-off between Precision and Recall or any of the other metrics that are of interest can be chosen. Figure 4 illustrates this point with two examples. The Precision and Recall shown in the two panes were calculated on the test examples predicted Active which exceeded both a Credibility threshold and a given Confidence threshold. In the left panel, the Credibility threshold was fixed and the Confidence threshold was varied; vice versa in the right panel.

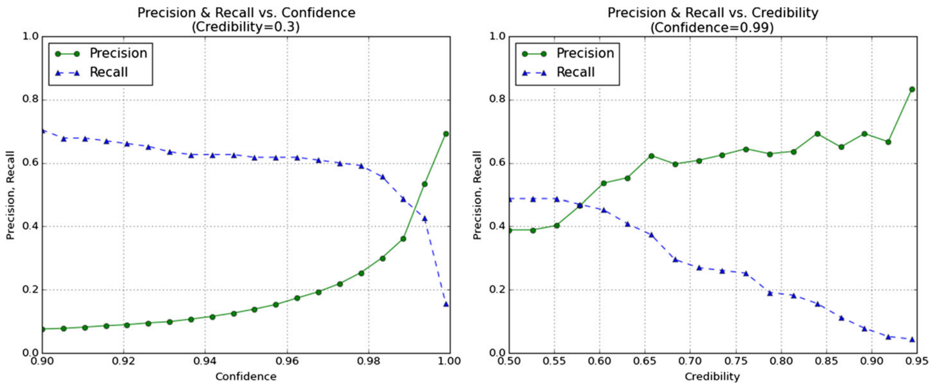

Fig. 4 Trade-off between precision and recall by varying credibility or confidence

3.4 Application to different data sets

We applied Inductive Conformal Predictors with underlying SVM using Tanimoto+RBF kernel to other data sets extracted from PubChem BioAssay to verify if the same performance would be achieved for assays covering a range of quite different biological targets and to what extent the performance would vary with differences in training set size, imbalance, and sparseness of the training set. The main characteristics of the data sets are reported in Table 7.

As in the previous set of experiments, 20 cycles of training and testing were performed and the results averaged over them. In each cycle, a test set of 10,000 examples was set aside and the rest was split between calibration set ($\approx 30,000$) and proper training set. The results are reported in Table 8.

It can be seen that the five data sets differ in their hardness for machine learning and in particular some of them produce more uncertain predictions using the same algorithms, number of examples and the same significance level.

3.5 Mondrian ICP with different ϵ_{active} and $\epsilon_{inactive}$

When applying Mondrian ICP, there is no constraint to use the same significance ϵ for the two labels. Using two different significance levels allows to vary the relative importance of

Table 7 Data sets and their characteristics

Data set	Assay description	Number of compounds	Number of features	Actives (%)	Density (%)
827	High Throughput Screen to Identify Compounds that Suppress the Growth of Cells with a Deletion of the PTEN Tumor Suppressor	138,287	165,786	1.2%	0.034%
1461	qHTS Assay for Antagonists of the Neuropeptide S Receptor: cAMP Signal Transduction	208,069	211,474	1.11%	0.026%
1974	Fluorescence polarization-based counterscreen for RBBP9 inhibitors: primary biochemical high throughput screening assay to identify inhibitors of the oxidoreductase glutathione S-transferase omega 1(GSTO1)	302,310	237,837	1.05%	0.024%
2553	High throughput screening of inhibitors of transient receptor potential cation channel C6 (TRPC6)	305,308	236,508	1.06%	0.024%
2716	Luminescence Microorganism Primary HTS to Identify Inhibitors of the SUMOylation Pathway Using a Temperature Sensitive Growth Reversal Mutant Mot1-301	298,996	237,811	1.02%	0.024%

Density refers to the percentage of non-zero entries in the full matrix of ‘Number of Compounds × Number of Features’ elements

Table 8 Results of the application of Mondrian ICP with $\epsilon = 0.01$ using SVM with Tanimoto+RBF as underlying

DataSet	Active pred Active	Inactive pred Active	Inactive pred Inactive	Active pred Inactive	Empty pred	Uncertain
827	47.65	94.10	1044.90	0.95	0	8812.40
1461	29.45	101.30	1891.10	1.20	0	7976.95
1974	62.50	97.40	880.85	1.00	0	8958.25
2553	34.00	101.00	337.90	1.00	0	9526.10
2716	3.55	98.20	97.00	1.00	0	9800.25

Test set size: 10,000

the two kinds of mis-classifications. Indeed, there may be an advantage in allowing different "error" rates for the two labels especially when the focus might be on identifying Actives rather than Inactives.

The validity of Mondrian machines implies that the expected number of certain but wrong predictions is bounded by ϵ_{act} for (true) actives and by ϵ_{inact} for (true) non-actives.

Figure 5 shows the trade-off between Precision and Recall that results from varying ϵ_{inact}.

For very low values of the significance ϵ, a large number of test examples have $p_{act} > \epsilon_{act}$ as well as $p_{inact} > \epsilon_{inact}$. For these test examples, we have an 'Uncertain' prediction.

As we increase ϵ_{inact}, fewer examples have a p_{inact} larger than ϵ_{inact}. So 'Inactive' is not chosen any longer as a label for those examples. If they happen to have a $p_{act} > \epsilon_{act}$,

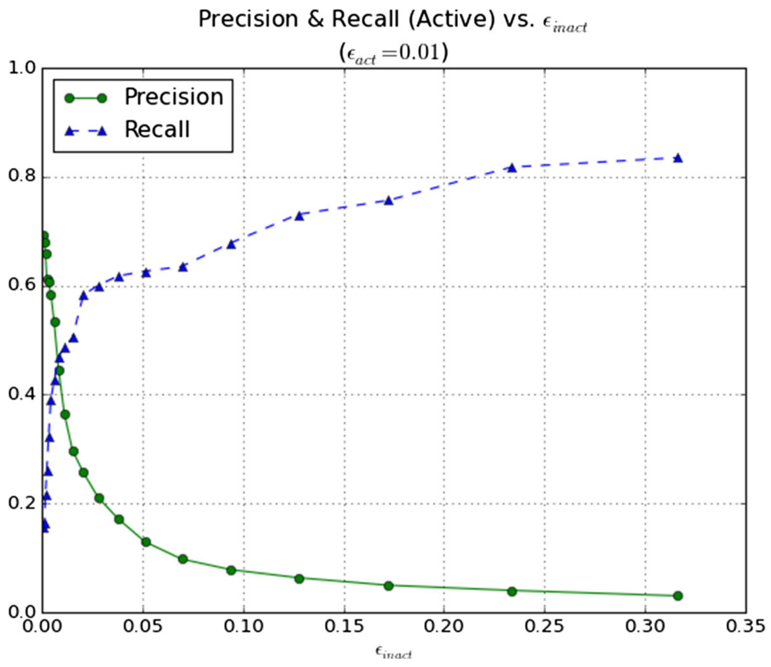

Fig. 5 Trade-off between precision and recall by varying ϵ_{inact}

they switch from ‘Uncertain’ to being predicted as ‘Active’ (in the other case, they would become ‘Empty’ predictions).

Figure 6 shows how Precision varies with Recall using three methods: varying the threshold applied to the Decision Function of the underlying SVM, varying the significance ϵ_{inact} for the Inactive class, varying the credibility. The three methods appear to give similar results and might seem equivalent. In the next section, we discuss a perspective that is of practical interest and that is quite specific to the CP technique.

4 Ranking of compounds by p-value

A useful product of the application of Conformal Predictors is the ability to rank the compounds by the p-values. A similar approach was applied to protein-protein interactions in [21].

In this specific context, where the focus is primarily on identifying the compounds that are more likely to be active towards a given biological target, the desired ranking would naturally list the candidates by p_{active} in descending order (i.e. compounds with higher p-value would rank higher than compounds with lower p-value) and break any ties using the $p_{inactive}$ (in ascending order).

An example of the ranking that can be obtained in this way is illustrated in Table 9. The table lists the 20 compounds that were assigned the largest p_{active} values in one of the runs. As explained earlier, these are the compounds for which there was the largest proportion of calibration compounds for which the Non Conformity Measure was greater than or equal to the Non Conformity Measure associated to the hypothesis of the compound being active. This last sentence is not clear - we probably don’t need this sentence at all. We also need to say a bit more about good predictions we have made here.

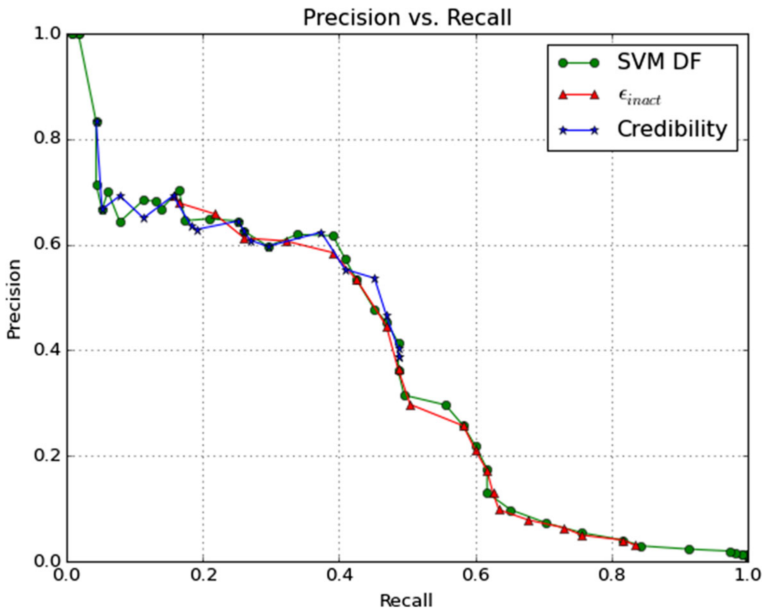

Fig. 6 Precision vs. recall: three methods

Table 9 The 20 candidate compounds (out of a held-out test set of 10,000) with the largest p_{active} values, from one of the runs for the data set AID827

Ranking	Compound ID	p_{active}	$p_{inactive}$
1	**108198**	**0.997067**	**0.000036**
2	103948	0.988270	0.000072
3	62772	0.988270	0.000143
4	129143	0.982405	0.000143
5	**108632**	**0.961877**	**0.000179**
6	**138051**	**0.961877**	**0.000179**
7	**108920**	**0.941349**	**0.000322**
8	**108877**	**0.938416**	**0.000322**
9	**108783**	**0.932551**	**0.000322**
10	**107957**	**0.932551**	**0.000322**
11	5413	0.926686	0.000358
12	4334	0.923754	0.000394
13	**138177**	**0.923754**	**0.000394**
14	71538	0.914956	0.000537
15	54806	0.914956	0.000537
16	16925	0.903226	0.000608
17	**108026**	**0.903226**	**0.000644**
18	**108584**	**0.900293**	**0.000644**
19	**107943**	**0.894428**	**0.000644**
20	**108032**	**0.894428**	**0.000644**

Bold face denotes those compounds that are actually Active. The compound ID is just the index in the pre-processed data set and not the PubChem ID

We believe this product of the application of CP is of particular practical interest because it allows the user (e.g. a pharmaceutical company) to select a set of promising compounds with a chosen level confidence.

5 Conclusions

This paper summarized a methodology of applying Mondrian Conformal Predictors to big and imbalanced data with several underlying machine learning methods like nearest neighbours, Bayesian, SVM and various kernels. The results have been compared from the point of view of efficiency of various methods and various sizes of the data sets. The paper also demonstrated how the error rate can be effectively controlled by changing the confidence level for the prediction. Lastly and perhaps most importantly, the paper provided an example of how Conformal Predictors can be used to rank compounds based on the confidence in their activity. Such ranking can be extremely useful in guiding the choice of the compounds to test, with the potential of reducing dramatically the investments required for identifying new candidate drugs.

The most interesting direction of the future extension is to study the possible strategies of active learning (or experimental design) and the practical problem of their integration into an on-going drug development process. The methods employed in this paper produce a number of uncertain predictions, in which both the Active and Inactive hypotheses cannot be rejected. It might be useful to select among those uncertain cases the compounds that should be checked experimentally first – in other words the most "promising" compounds. How to select though may depend on practical scenarios of further learning and on comparative efficiency of different active learning strategies.

Acknowledgments This project (ExCAPE) has received funding from the European Union's Horizon 2020 Research and Innovation programme under Grant Agreement no. 671555. We are grateful for the help in conducting experiments to the Ministry of Education, Youth and Sports (Czech Republic) that supports the Large Infrastructures for Research, Experimental Development and Innovations project "IT4Innovations National Supercomputing Center – LM2015070". This work was also supported by EPSRC grant EP/K033344/1 ("Mining the Network Behaviour of Bots") and by Technology Integrated Health Management (TIHM) project awarded to the School of Mathematics and Information Security at Royal Holloway as part of an initiative by NHS England supported by InnovateUK. We are indebted to Lars Carlsson of Astra Zeneca for providing the data and useful discussions. We are also thankful to Zhiyuan Luo and Vladimir Vovk for many valuable comments and discussions.

References

1. Monev, V.: Introduction to similarity searching in chemistry. Comm. Math. Comp. Chem. **51**, 7–38 (2004)
2. Bottou, L., Chapelle, O., DeCoste, D., Weston, J.: Large-scale kernel machines (neural information processing). The MIT press (2007)
3. Bussonnier, M.: Interactive parallel computing in Python. https://github.com/ipython/ipyparallel
4. Pérez, F., Granger, B.E.: IPython: a system for interactive scientific computing, vol. 9 (2007). http://ipython.org
5. Kluyver, T., et al.: Jupyter Notebooks – a publishing format for reproducible computational workflows. Positioning and Power in Academic Publishing: Players, Agents and Agendas, 87–90 doi:10.3233/978-1-61499-649-1-87
6. Chang, C.-C., Lin, C.-J.: LIBSVM: A library for support vector machines. ACM Trans. Intell. Syst. Technol. **2**, 27:1–27:27 (2011). Software available at http://www.csie.ntu.edu.tw/~cjlin/libsvm
7. Chang, E.Y.: PSVM: parallelizing support vector machines on distributed computers. In: Foundations of Large-Scale Multimedia Information Management and Retrieval, pp. 213–230. Springer, Berlin Heidelberg (2011)
8. Faulon, J.-L., Visco, D.P. Jr., Pophale, R.S.: The signature molecular descriptor. 1. using extended valence sequences in qsar and qspr studies. J. Chem. Inf. Comput. Sci. **43**(3), 707–720 (2003). PMID: 12767129
9. Gammerman, A., Vovk, V.: Hedging predictions in machine learning. Comput. J. **50**(2), 151–163 (2007)
10. Gärtner, T.: Kernels for Structured Data. World Scientific Publishing Co., Inc., River Edge (2009)
11. Graf, H.P., Cosatto, E., Bottou, L., Durdanovic, I., Vapnik, V.: Parallel support vector machines: the cascade SVM. In: Advances in Neural Information Processing Systems, pp. 521–528. MIT Press (2005)
12. Jain, A.N., Nicholls, A.: Recommendations for evaluation of computational methods. J. Comput. Aided Mol. Des. **22**(3-4), 133–139 (2008)
13. 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, É.: Scikit-learn: Machine learning in Python. J. Mach. Learn. Res. **12**, 2825–2830 (2011)
14. Shafer, G., Vovk, V.: A tutorial on conformal prediction. J. Mach. Learn Res. **9**, 371–421 (2008)
15. Vovk, V., Gammerman, A., Shafer, G.: Algorithmic Learning in a Random World. Springer-Verlag New York, Inc., Secaucus, NJ, USA (2005)
16. Weis, D.C., Visco, D.P. Jr.: Jean-loup Faulon. Data mining pubchem using a support vector machine with the signature molecular descriptor Classification of factor {XIa} inhibitors. J. Mol. Graph. Model. **27**(4), 466 –475 (2008)
17. Holenz, J., et al. (eds.): Lead Generation: Methods and Strategies, vol. 68. Wiley-VCH (2016)
18. Woodsend, K., Gondziom, J.: Hybrid MPI/OpenMP parallel linear support vector machine training. J. Mach. Learn. Res. **10**, 1937–1953 (2009)

19. You, Y., Fu, H., Song, S.L., Randles, A., Kerbyson, D., Marquez, A., Yang, G., Hoisie, A.: Scaling support vector machines on modern HPC platforms. J. Parallel Distrib. Comput. **76**(C), 16–31 (2015)
20. Toccaceli, P., Nouretdinov, I., Gammerman, A.: Conformal predictors for compound activity prediction. In: COPA Proceedings of the 5th International Symposium on Conformal and Probabilistic Prediction with Applications, vol. 9653, p. 2016. Springer-Verlag New York Inc. (2016)
21. Nouretdinov, I., Gammerman, A., Qi, Y., Klein-Seetharaman, J.: Determining confidence of predicted interactions between HIV-1 and human proteins using conformal method. Pac. Symp. Biocomput. 311 (2012)
22. Wang, Y., Suzek, T., Zhang, J., Wang, J., He, S., Cheng, T., Shoemaker, B.A., Gindulyte, A., Bryant, S.H.: Pubchem BioAssay: 2014 upyear. Nucleic Acids Res. **42**(1), D1075–82 (2014)
23. McCool, M., Robison, A.D., Reinders, J.: Structured Parallel Programming: Patterns for Efficient Computation. Morgan-Kaufmann (2012)