The Polypharmacology Browser PPB2: Target Prediction Combining Nearest Neighbors with Machine Learning

Mahendra Awale\textsuperscript{a)} and Jean-Louis Reymond\textsuperscript{a)*}

\textsuperscript{a)} Department of Chemistry and Biochemistry, National Center of Competence in Research NCCR TransCure, University of Berne, Freiestrasse 3, 3012 Berne Switzerland

E-mail: jean-louis.reymond@dcb.unibe.ch

Abstract

Here we report PPB2 as a target prediction tool assigning targets to a query molecule based on ChEMBL data. PPB2 computes ligand similarities using molecular fingerprints encoding composition (MQN), molecular shape and pharmacophores (Xfp), and substructures (ECfp4), and features an unprecedented combination of nearest neighbor (NN) searches and Naïve Bayes (NB) machine learning, together with simple NN searches, NB and Deep Neural Network (DNN) machine learning models as further options. Although NN(ECfp4) gives the best results in terms of recall in a 10-fold cross-validation study, combining NN searches with NB machine learning provides superior precision statistics, as well as better results in a case study predicting off-targets of a recently reported TRPV6 calcium channel inhibitor, illustrating the value of this combined approach. PPB2 is available to assess possible off-targets of small molecule drug-like compounds by public access at ppb2.gdb.tools.
**Introduction**

In ligand-based virtual screening (LBVS) one uses structural similarities between molecules to infer possible similarities in their biological activities. The approach is used broadly to accelerate and reduce the costs of the initial steps of drug discovery by guiding experimental screening to molecules with a higher probability of activity. LBVS also forms the basis for target prediction tools, which assign possible targets to a molecule based on its similarity to known, target annotated molecules such as those in the ChEMBL database. Targets are assigned either directly based on nearest neighbor (NN) relationships, or indirectly by building a machine learning (ML) model, with several tools available online.

Herein we report PPB2 (Polypharmacology Browser version 2) as an extension of our previously reported web-portal PPB (Polypharmacology Browser). While PPB predicted targets by simple NN relationships to known compounds from ChEMBL using multiple molecular fingerprints simultaneously, PPB2 uses only three molecular fingerprints, but exploits them in a combination of nearest neighbor (NN) searches and Naïve Bayes (NB) machine learning, which to the best of our knowledge is unprecedented for target prediction. Simple NN searches as well as NB and Deep Neural Network (DNN) machine learning models are also available in PPB2 as further options. While NN(ECfp4) is the best method in terms of recall in a 10-fold cross-validation study, we find that the NN+NB combinations perform best in terms of precision statistics. The combination methods also stand out in a case study predicting off-targets of a recently reported TRPV6 calcium channel inhibitor, where they correctly predict hERG as an important off-target missed by NN searches as well as by our previous tool PPB and by many other online target prediction tools. PPB2 is freely available for use at ppb2.gdb.tools.
Results and Discussion

To build PPB2 we collected a bioactivity dataset of all compounds having at least IC$_{50}$ < 10 $\mu$M on a single protein target in ChEMBL22,\textsuperscript{35} considering only high confidence data points as annotated in ChEMBL and only targets for which at least 10 compounds were documented. This provided 344,163 single compounds associated with 1,720 single protein targets belonging to 8 different target families, representing 555,346 target-compound associations.

To encode molecular structures we selected three fingerprints perceiving different levels of details, namely: 1) MQN (Molecular Quantum Number), a 42-bit scalar fingerprint representing molecular composition by atoms, bonds, polar groups and topological features\textsuperscript{36,37} particularly useful to search and visualize large databases;\textsuperscript{38-40} 2) Xfp (atom category extended atom-pair fingerprint), an 55-bit scalar fingerprint perceiving molecular shape and pharmacophores and well suited for scaffold-hopping virtual screening;\textsuperscript{41} and 3) ECfp4 (extended connectivity fingerprint up to four bonds), a 1024-bit binary substructure fingerprint encoding detailed information about molecular structure.\textsuperscript{42} Similarities were computed using the city-block distance for MQN and Xfp and the Tanimoto coefficient for ECfp4.\textsuperscript{43}

In terms of search methods, we first implemented simple NN searches with each of these three fingerprints because the method is straightforward and often gives satisfactory results. Secondly, we implemented our key idea for PPB2, which was to build a specific ML model for each query molecule on the fly using NN compounds of the query. This approach would avoid making predictions based on non-realistic associations between compounds that would be structurally too distant from each other, thus capitalizing on both the “no-nonsense” advantage of NN searches and the deeper exploitation of multiple datapoints possible with ML. The best combination involved retrieving the 2,000 NN of a query molecule using MQN, Xfp, or ECfp4, followed by building a Laplacian modified Multinomial NB model on the fly from these 2,000 NN to provide the actual target prediction. The number of 2,000 NN was large enough to include all high-similarity
compounds and small enough to enable a fast NB model building on the fly. Finally, we also implemented a direct NB model and a deep neural network (DNN) model trained with the entire dataset using ECfp4. In total this provided eight different search methods for PPB2 (Figure 1a).

In the PPB2 web-portal the user inputs a molecule as either SMILES or structural drawing and selects a prediction method. Prediction results are shown as a list of the most probable targets (Figure 1b). For each target the ChEMBL molecules on which the target prediction is based can be inspected visually by opening the “Show NN” window (Figure 1c). Molecular structures are drawn from SMILES within the browser using SmilesDrawer.44

![Figure 1](image_url)

**Figure 1.** The polypharmacology browser PPB2. (a) Workflow of the four different target prediction methods. Red arrows: retrieve top 2,000 NN using MQN (city block distance) or Xfp (city block distance) or ECfp4 (tanimoto coefficient) and rank the targets as per the similarity score of the most similar nearest neighbor associated with each target. Blue arrows: retrieve top 2,000 NN using MQN or Xfp or ECfp4, then build on the fly an ECFP4-Naive Bayes (NB) machine learning model based on these 2,000 nearest neighbors and perform the target prediction. Orange arrows: perform target prediction using an ECFp4-Naive Bayes machine learning model built on ChEMBL. Green arrows: perform target prediction using an ECFp4-Deep Neural Network (DNN) machine learning model built on ChEMBL. (b) Query molecule entry window in PPB2 with example molecule CIS22a. (c) PPB2 target prediction results window showing rank, ChEMBL ID and common name for predicted targets, exemplified with compound CIS22a and the NN(ECfp4) + NB(ECfp4) method. (d) Display of the first four NNs associated with hERG. This panel opens through the “Show NN” green button in the target list (c).
The performance of the eight different target prediction methods in PPB2 was analyzed by calculating recall and precision statistics of a 10-fold cross-validation study. The performance in terms of overall recall (percentage of known compound-target pairs in the test set that are predicted by PPB2) increased from 47-61 % when considering only the top predicted target to 85-96 % when considering the top 10 predicted targets, while at the same time the overall precision (percentage of compound-target pairs predicted by PPB2 which correspond to known interactions) decreased from 69-91 % to 13-14 % (Figure 2a, left panels). Recall was lower but precision higher with a similar trend when calculated on a per target basis, which compensates for the fact that some targets are overrepresented (Figure 2a, right panels). Similar results were obtained when considering only compounds associated with a single protein target (Figure 2b). Performance was most strongly influenced by the ECfp4 Tanimoto similarity between the query molecule and its ECfp4 NN in the training set, with a drastic reduction of both recall and precision when the value dropped below 0.5 (Figure 2c/d). Performance was also influenced by the target class, with membrane receptors performing best and kinases performing worst, an effect also observed with other target prediction tools (Figure 2e).13

Across the different comparisons above, the simple NN(ECfp4) methods performed best in terms of recall statistics, which has also been noted by other authors in a related target prediction study.17 In terms of precision statistics by contrast, there was a clear advantage for the combined NN+NB methods, in particular when computed on a per target basis, which is an important parameter as it gives an indication of the expected rate of success for target assignment (lower right panels in Figure 2a-d and Figure 2e). On the other hand the simple NN(MQN) and the NB(ECfp4) methods gave relatively poorer results across all comparisons, showing that encoding compounds by composition only (MQN) or using a NB model alone (even with ECfp4) does not exploit the target-compound pair information as well as the other, more complex methods in PPB2 for target prediction.
Figure 2. Performance of PPB2 in a 10-fold cross-validation (see method for details). (a) Overall (all ligand-target pairs) and average (per target) recall and precision performance as function of the number or targets considered. (b) Same as (a) considering only compounds annotated with a single target (241,083 compounds, ~70% of database). (c) Overall and average recall and precision considering top 5 predicted targets as function of ECfp4 tanimoto coefficient similarity of the compound in the test set to its NN in the training set. (d) Same as (c) for single target compounds only. (e) Average performance (per target) estimated per target class considering the top 5 predicted targets.
As an application example we analyzed the case of compound CIS22a, a recently reported TRPV6 calcium channel inhibitor for which we had measured 24 possible off-targets from the “safety screen” panel of Cerep Pvt. Ltd. We compared the prediction of PPB2 with those of SwissTargetPrediction, SEA, Modlab Spider, SuperPred, HitPick, TargetHunter, Chemmapper, TarPred as well as our previously reported PPB, considering for each tool the top 20 predicted targets (Figure 3, the structure of CIS22a is shown in Figure 1).

PPB2 predicted 6-8 of the 12 off-targets against which CIS22a is active, and only 1-2 of 12 possible off-targets against which CIS22a is not active. Most interestingly, the three combined NN+NB(ECfp4) methods as well as the NB(ECfp4) and DNN(ECfp4) correctly predicted hERG, an important off-target of CIS22a which was missed by the simple NN searches, illustrating the value of ML approaches. Note that the combined NN+NB methods as well as the DNN did not predict any of the inactive off-targets, while the NB method incorrectly predicted one of the inactive off-targets.

Among the other search tools, only SPIDER and TarPred predicted hERG for CIS22a. TarPred also correctly predicted five other off-targets and did not predict any of the inactive off-targets. SPIDER predicted 11 out of the 12 off-targets of CIS22a including hERG, but also predicted 5 out of the 12 inactive off-targets, suggesting that this tool favors recall over precision. The other tools tested failed to predict hERG but did not predict any of the inactive off-targets, indicating that they favor precision over recall.
Figure 3. Target prediction results for the eight prediction methods in PPB2 and other prediction tools available online for the TRPV6 inhibitor CIS22a. The structure of CIS22a is shown in Figure 1. For each option in PPB2 and each online tool the 20 top predicted targets were considered. Target full names: Voltage gated potassium channel subfamily H member 2 (HERG), Adrenergic α1A (ADRA1A) and α2A (ADRA2A) receptor, Dopamine receptor subtypes D1-D4 (DRD1-4), Cholinergic muscarinic receptor 1 (CHRM1) and 2 (CHRM2), µ opioid receptor (OPRM), 5-Hydroxytryptamine receptor 1A (HTR1A), 1B (HTR1B), 2A (HTR2A) and 2B (HTR2B), Voltage gated Na+ channel (SCN2A), Voltage dependent L- (CACNA1S) and N-type (CACNA1B) Ca2+ channel, 5-Hydroxytryptamine receptor 3 (5-HT3), N-methyl-D-aspartate receptor (NMDA), Gamma aminobutyric acid receptor (GABA), Adrenergic β1 (ADRB1) and β2 (ADRB2) receptor, Cannabinoid 1 (CB1) and 2 (CB2) receptor.

Conclusion

The key idea for the target prediction tool presented here was to combine nearest neighbor (NN) searches with Naive Bayes (NB) machine learning to enhance performance. We performed similarity calculations based on three different molecular fingerprints, namely MQN as a composition fingerprint, Xfp as a shape and pharmacophore fingerprint, and ECfp4 as a circular substructure fingerprint. Eight different search options featuring NN searches alone, NN searches combined with NB, as well as NB and DNN methods, were incorporated into PPB2 as a web-based target prediction tool. While NN(ECfp4) performed best in terms of recall statistics in the cross-validation study, the combined NN+NB methods showed superior performance in terms of precision and returned better results in a case study with the TRPV6 calcium channel inhibitor CIS22a, in particular regarding prediction of hERG as off-target. PPB2 is freely available at
and can be used to assess possible off-targets of small molecule drug-like compounds.

**Methods**

**Cheminformatics**

*Dataset.* All compound-target interactions were extracted from the publicly available MySQL version of the ChEMBL database (release 22). Criteria for selecting interactions from the database were targets labeled as “single protein” with the source organism being either human or rat and compounds associated with the target were to have IC$_{50}$, EC$_{50}$, $K_i$ or $K_D$ value of $\leq 10 \, \mu$M, confidence score $> 5$ and heavy atom count $\leq 50$. In a next step, compounds were processed in non-isomeric SMILES format, where counter ions were removed, valence errors corrected, and the compounds ionized at pH 7.4. Duplicate molecules were then removed based on unique SMILES comparison in the context of each target protein, and targets with less than 10 compounds were discarded. Filtering the database using these criteria resulted in a subset containing 1,720 target proteins, 344,163 unique compounds and 555,346 compound-target interactions. This subset was stored in a plain text file (henceforth referred to as the database) grouped by SMILES resulting in 344,163 lines, with each line containing a compound encoded as a SMILES string followed by the associated target names.

*Fingerprints.* For each of compounds in the database we calculated the MQN (42D), Xfp (55D) and ECfp4 (1024D) fingerprints. All fingerprint calculations were performed using an in-house java program utilizing various plugins from the ChemAxon JChem library such as MajorMicrospeciesPlugin to adjust the ionization state of molecules, HBDAPlugin to determine hydrogen bond donor and acceptor atoms and TopologyAnalyserPlugin to determine the shortest topological path between atom-pairs.
Nearest neighbor (NN) search. For a given query molecule NN searches were performed in the training set using MQN, Xfp or ECfp4. For MQN and Xfp the similarity between query (A) and a compound in the training set (B) was calculated using city block distance (CBD) as follows, where.

\[
CBD(A, B) = \sum_{i=1}^{n} |A_i - B_i|
\]

For the ECfp4 fingerprint the similarity between compounds was calculated using the Tanimoto coefficient (T) as follows, where A = ECfp4 fingerprint of the query molecule, B = ECfp4 fingerprint of a compound in the training set, \(N_C\) = number of ON bits common between molecule A and B, \(N_A\) and \(N_B\) = number of ON bits in the query and a compound in training set, respectively:

\[
T(A, B) = \frac{N_C}{N_A + N_B - N_C}
\]

After the similarity calculation compounds in the training set were sorted with respect to the query molecule (high to low similarity: increasing CBDs or decreasing Tanimoto coefficient) and the top 2,000 compounds were extracted.

Naïve Bayes Model. A Multinomial Naïve Bayes Classifier (NB) model with Laplacian smoothing was trained using the Python-based scikit-learn (version 0.19) machine learning library.

Fingerprints of molecules were used as input feature vectors, while target names were used as class labels. NB classifier is probabilistic in nature, which means that for any given query compound the NB classifier calculates the probability for each target present in the training set and considers the targets with the highest probabilities (top \(N\) targets) as the predicted targets of a query molecule. This probability calculation is fundamentally based on the Bayesian theorem of conditional probability, wherein the probability of an event A happening considering an evidence B (\(P(A|B)\)) can be calculated based on prior probability of an evidence B in the sample of an event A (\(P(B|A)\))
and prior probability of an event A \( (P(A)) \) in the training set. For target prediction \( (A = \text{target}, B = \text{fingerprint of a molecule}) \) the probability \( P(A|B) \) of a compound to be active on target A is calculated from \( (P(B|A)) = \text{prior probabilities calculated based on fingerprints of known bioactive compounds active against target A in the training set, } P(A) = \text{the relative frequency of target A considering all the other targets in the training set, and } P(B) = \text{the probability of evidences across the entire training set, as follows:} \)

\[
P(A|B) = \frac{P(B|A) \cdot P(A)}{P(B)}
\]

The NB classifier uses the naïve assumption that all the features (evidences) are independent from each other and contribute independently to probability calculation. To calculate \( P(A|B) \) given multiple evidences \( (B = B_{i=1} \ldots B_{n}) \) one needs to estimate the product of the probability of each evidence given an event A \( (P(B|A) = P_{B1A} \cdot P_{B2A} \cdots P_{nA}) \) and multiplied by the prior probability of an event A in the training set. The probability for each of the evidences given event A is True \( (P_{BIA}) \) can be calculated as follows, where \( N_{iA} = \text{count of } i^{th} \text{ evidence in a sample of event A in the training set, } N_A = \text{total count of all evidences in a sample of an event A in the training set, } n = \text{number of evidences and } \alpha = \text{smoothing factor:} \)

\[
P_{BIA} = \frac{N_{iA} + \alpha}{N_A + \alpha n}
\]

When \( \alpha = 1 \), the calculation is called Laplace smoothing. Parameter \( \alpha \) is useful to avoid zero probability resulting from the absence of the evidences in the sample set. Further technical details about the NB classifier can be found in the Scikit-learn documentation and corresponding GitHub repository (https://github.com/scikit-learn/scikit-learn/blob/master/sklearn/naive_bayes.py)

**Deep Neural Network Model (DNN).** The DNN model was built using Keras (version 2.0.9), a python based deep learning library with Tensorflow-gpu backend. The DNN model presented herein contains a 1,024D input layer corresponding to 1,024D ECfp4 fingerprints of molecules, two hidden layers containing 1,000 and 500 neurons respectively, and a 1,720D binary vector as output
layer (1: compound active against target, 0: compound not active against target) corresponding to 1,720 targets in database. Rectified linear units (relu) were used as activation functions for the hidden layers with a drop-out rate of 20% to avoid model-overfitting, while a sigmoid activation function was used for the output layer. The model was trained using “adam” as optimizer and binary crossentropy as loss function. The numbers of epoch iterations and the batch size were set to 200 and 500 respectively. Initially, series of models were built and evaluated in 10-fold cross validation with progressive optimization of number of neurons and number of hidden layers in network.

**Cross-validation study**

*Training and Test set.* The records (lines) in the database were randomized and subdivided into 10 distinct sets of equal size. Each of these 10 sets was then used as a test set with the remaining 9 sets being combined into the training set.

*Recall and Precision.* The target prediction performance of the different methods (NN(MQN) + NB(ECfp4), NN(Xfp) + NB(ECfp4), NN(ECfp4) + NB(ECfp4), NN(MQN), NN(Xfp), NN(ECfp4), NB(ECfp4) and DNN(ECfp4)) was measured using recall and precision parameters in a 10-fold cross validation study. We carried out target prediction only for compounds associated with up to 10 targets. For each compound we considered the top *n* predicted targets and evaluated whether the annotated targets for the compound could be recalled by the prediction.

For each target *x* we calculated a) true positives (TP): the number of times target *x* was correctly predicted; b) false negatives (FN): the number of interactions of target *x* which were not predicted; c) false positives (FP): the number of times target *x* was wrongly predicted (found among top *n* predicted targets but not annotated for the compound).

We calculated overall recall, average recall, overall precision and average precision as follows: **Overall recall:** The number of true positives divided by the number of true positives plus
the number of false negatives across all the targets \((i)\) and all the 10 cross-validation runs \((j)\) as shown in equation 1. *Average recall:* The average recall for each target across 10 cross-validation runs (equation 2). Similarly, the overall precision (equation 3) and average precision (equation 4) were calculated. Furthermore, targets were classified as per target class and overall recall, average recall, overall precision and average precision were calculated class-wise.

\[
\text{Overall recall} = \frac{\sum_{i=1}^{n} \sum_{j=1}^{10} TP_{i,j}}{\sum_{i=1}^{n} \sum_{j=1}^{10} TP_{i,j} + \sum_{i=1}^{n} \sum_{j=1}^{10} FN_{i,j}} \cdot 100
\] (1)

\[
\text{Average recall} = \frac{1}{n} \sum_{i=1}^{n} \left( \frac{\sum_{j=1}^{10} TP_{i,j}}{\sum_{j=1}^{10} TP_{i,j} + \sum_{j=1}^{10} FN_{i,j}} \right) \cdot 100
\] (2)

\[
\text{Overall precision} = \frac{\sum_{i=1}^{n} \sum_{j=1}^{10} TP_{i,j}}{\sum_{i=1}^{n} \sum_{j=1}^{10} TP_{i,j} + \sum_{i=1}^{n} \sum_{j=1}^{10} FP_{i,j}} \cdot 100
\] (3)

\[
\text{Average precision} = \frac{1}{n} \sum_{i=1}^{n} \left( \frac{\sum_{j=1}^{10} TP_{i,j}}{\sum_{j=1}^{10} TP_{i,j} + \sum_{j=1}^{10} FP_{i,j}} \right) \cdot 100
\] (4)

Where \(TP = \text{true positives}, FP = \text{false positives}, FN = \text{false negatives}, i = \text{target}, j = \text{cross validation run}\)

**Authors’ contributions.** MA designed, realized and implemented PPB2 and wrote the paper. JLR designed supervised the study and wrote the paper.

**Notes.** The authors declare no competing financial interest.

**Acknowledgments**

This work was supported financially by the University of Berne, the Swiss National Science Foundation and the NCCR TransCure. We thank ChemAxon Pvt. Ltd. for providing free academic and web licenses for their products.
References


Graphics for the Table of Contents: