INSERM'S DAY: AI & MACHINE LEARNING FOR DRUG DISCOVERY & DEVELOPMENT

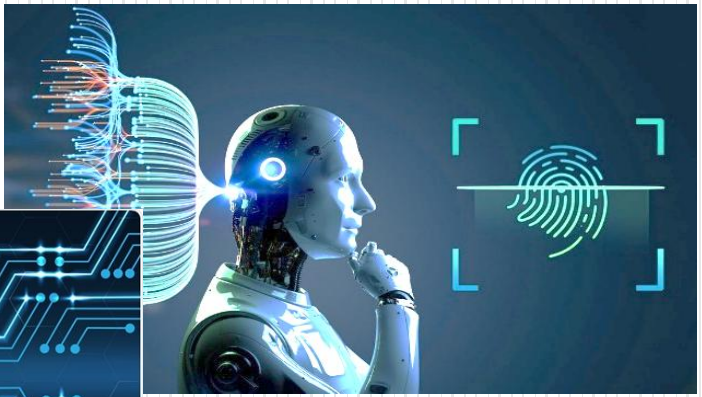
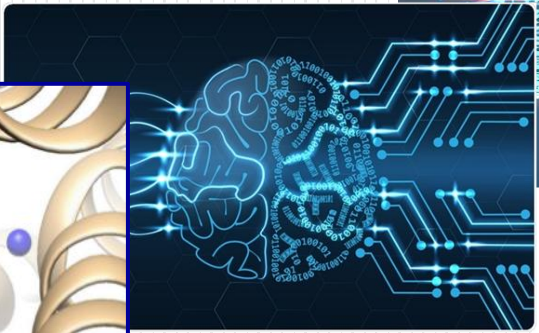
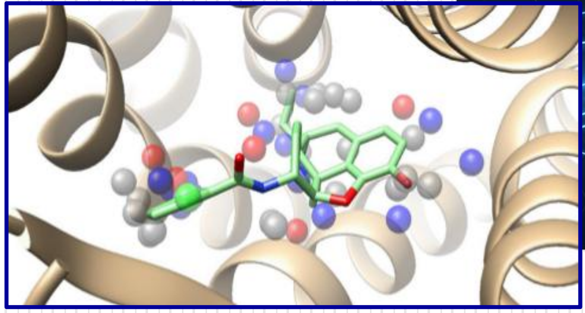
PHYSICS AND ARTIFICIAL INTELLIGENCE IN MOLECULAR DOCKING: FROM DOCKING POWER BENCHMARKING TO VIRTUAL SCREENING APPLICATIONS

Viet-Khoa Tran-Nguyen, Ph.D.

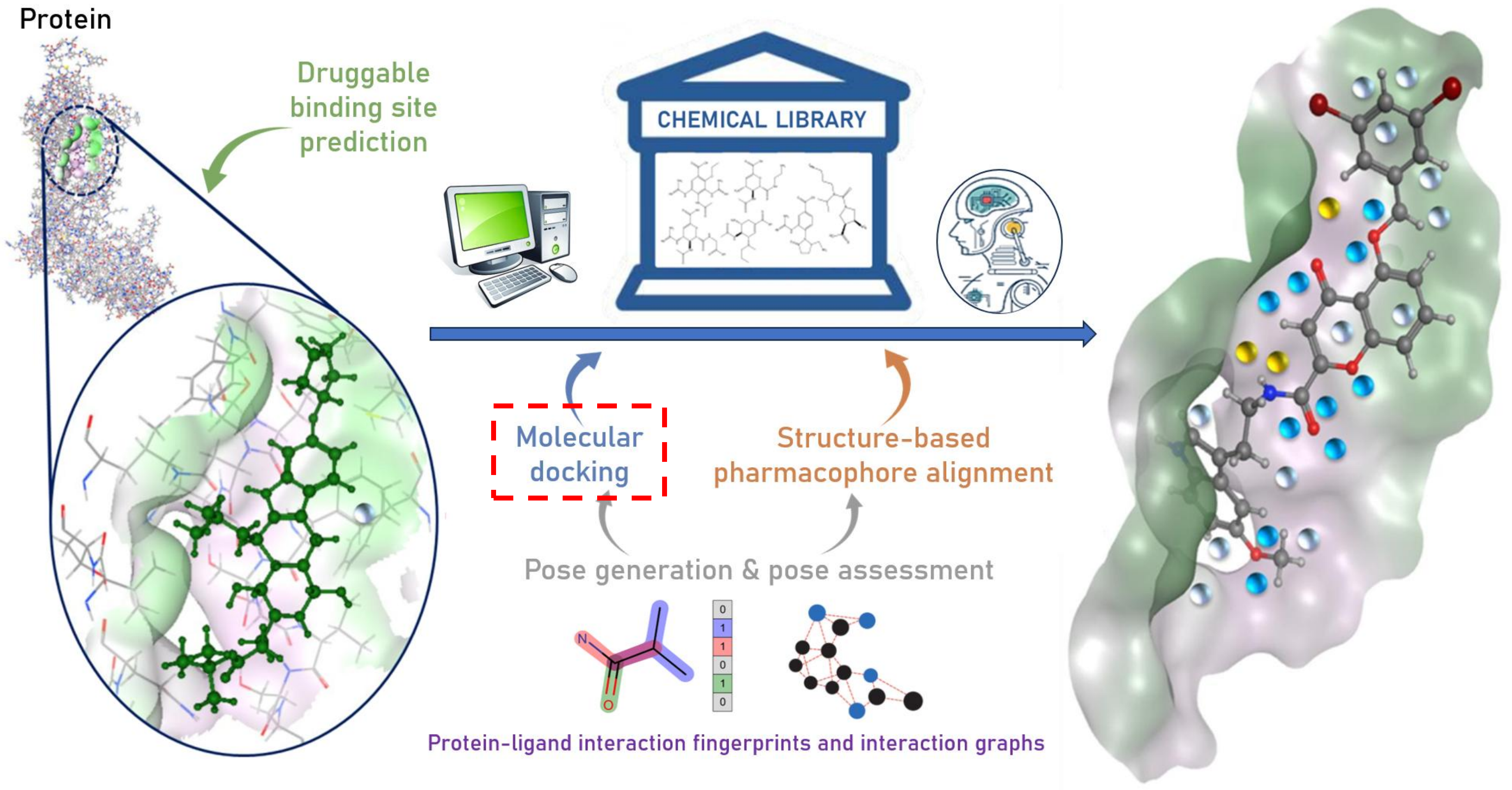
Assistant Professor (MCF)

Université Paris Cité

Paris, January 2026

INTRODUCTION


Protein
Druggable binding site prediction
CHEMICAL LIBRARY
Molecular docking
Structure-based pharmacophore alignment
Pose generation & pose assessment
N
O
0
1
1
0
1
0
Protein-ligand interaction fingerprints and interaction graphs

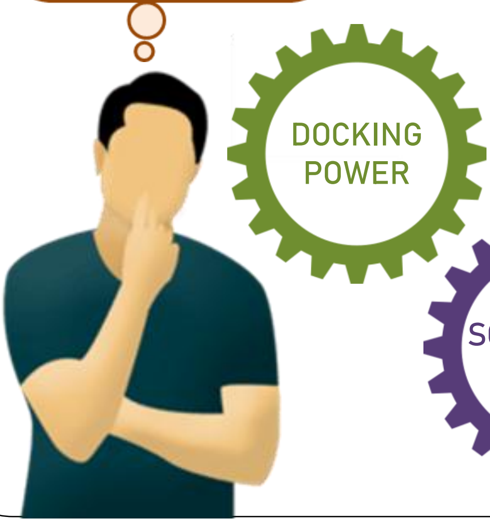
V.K. Tran-Nguyen *et al.*, *Nat. Protoc.* 2023, 18, 3460-511
V.K. Tran-Nguyen and A.C. Camproux, *Curr. Opin. Struct. Biol.* 2025, 95, 103152

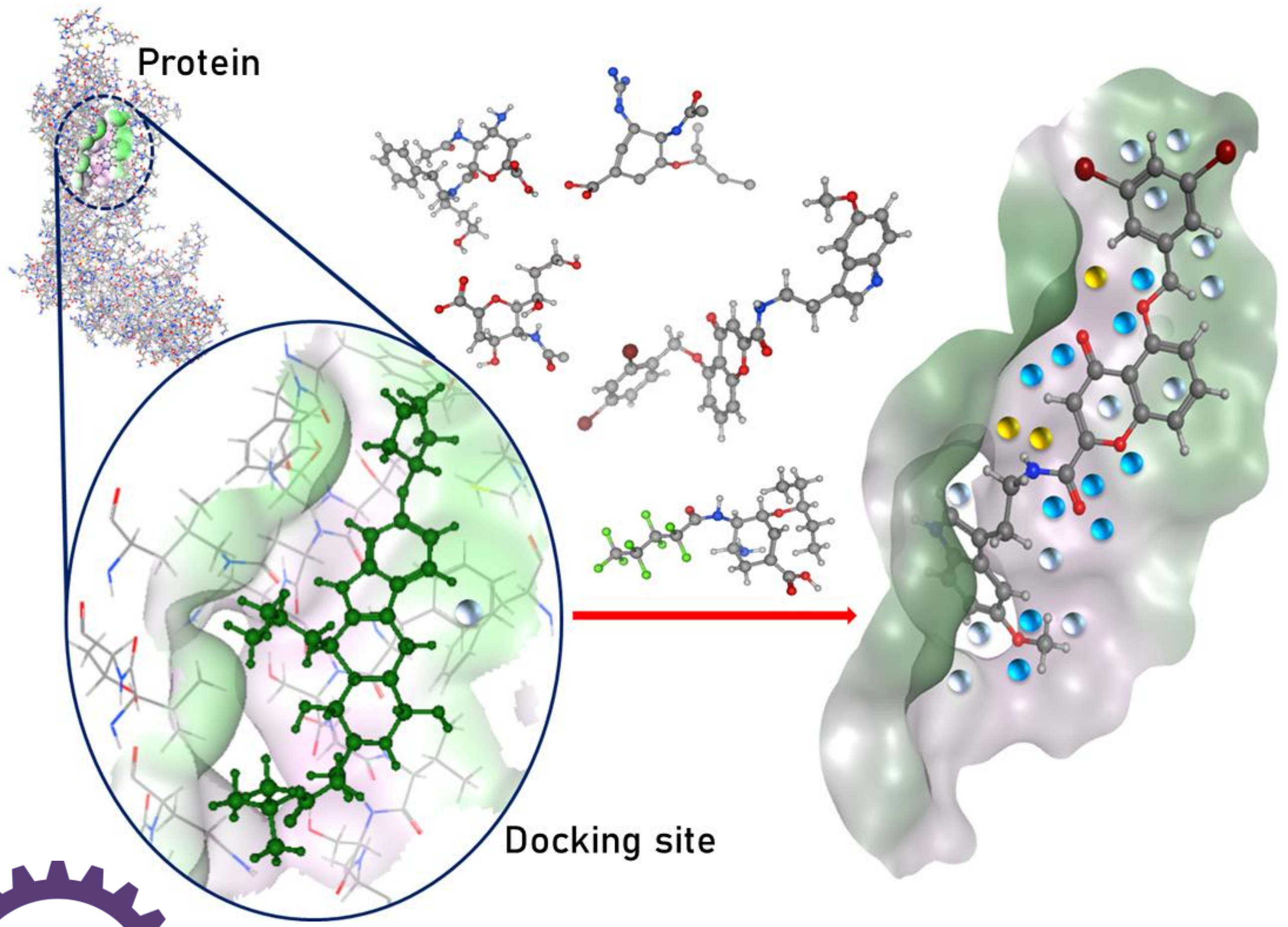
INTRODUCTION

Rigid or flexible side chains?

Number of poses to generate?

Pose selection strategy?

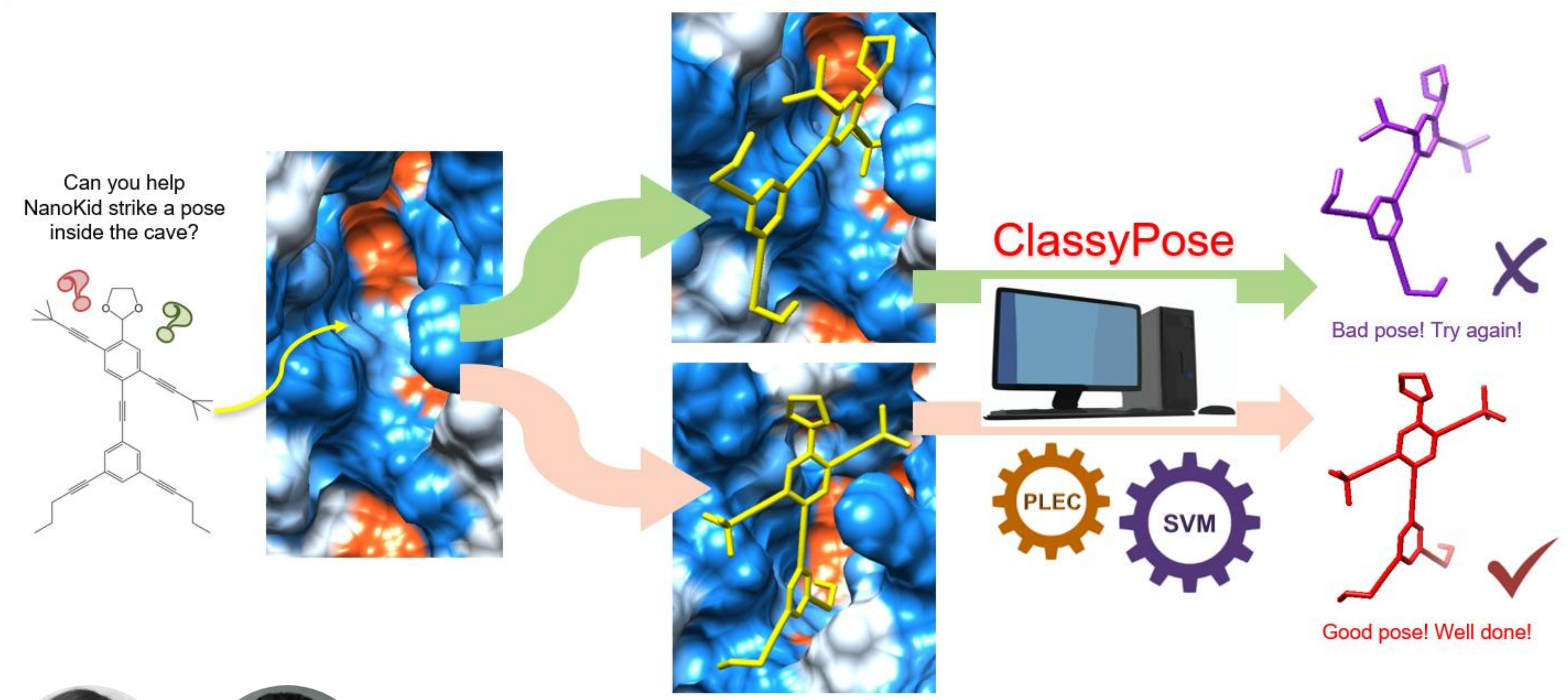




M. Su *et al.*, *J. Chem. Inf. Model.* 2019, 59, 895–913
B.J. Bender *et al.*, *Nat. Protoc.* 2021, 16, 4799-832
V.K. Tran-Nguyen *et al.*, *Nat. Protoc.* 2023, 18, 3460-511

DOCKING POWER: POSE GENERATION & SELECTION

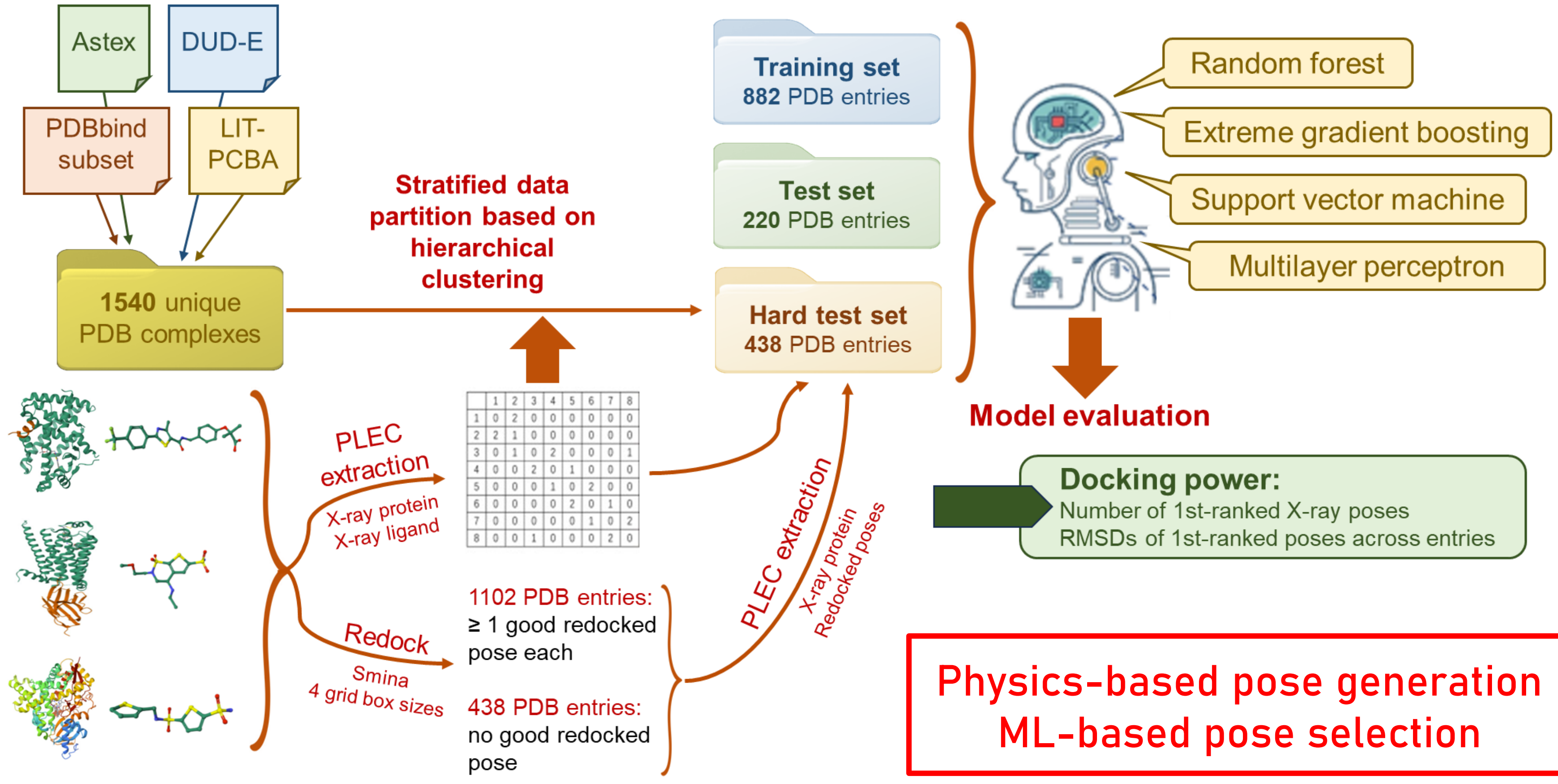
ClassyPose: a machine-learning classification model for ligand pose selection



V.K. Tran-Nguyen *et al.*, *Adv. Intell. Syst.* 2024, 6, 2400238

DOCKING POWER: POSE GENERATION & SELECTION

ClassyPose: a machine-learning classification model for ligand pose selection



	1	2	3	4	5	6	7	8
1	0	2	0	0	0	0	0	0
2	2	1	0	0	0	0	0	0
3	0	1	0	2	0	0	0	1
4	0	0	2	0	1	0	0	0
5	0	0	0	1	0	2	0	0
6	0	0	0	0	2	0	1	0
7	0	0	0	0	0	1	0	2
8	0	0	1	0	0	0	2	0

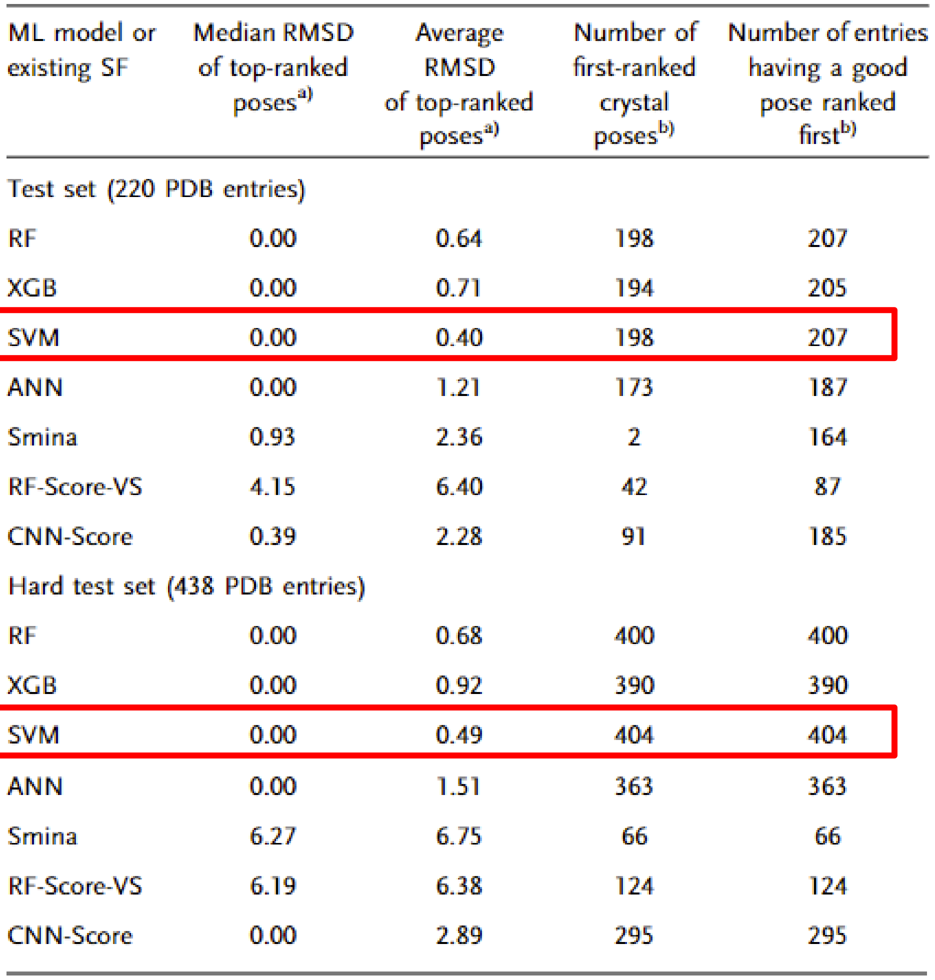
Docking power:
Number of 1st-ranked X-ray poses
RMSDs of 1st-ranked poses across entries

Physics-based pose generation
ML-based pose selection

V.K. Tran-Nguyen *et al.*, *Adv. Intell. Syst.* 2024, 6, 2400238
V.K. Tran-Nguyen *et al.*, *Nat. Protoc.* 2023, 18, 3460-511

DOCKING POWER: POSE GENERATION & SELECTION

ClassyPose: a machine-learning classification model for ligand pose selection



ML model or existing SF	Median RMSD of top-ranked poses[a)]	Average RMSD of top-ranked poses[a)]	Number of first-ranked crystal poses[b)]	Number of entries having a good pose ranked first[b)]
Test set (220 PDB entries)				
RF	0.00	0.64	198	207
XGB	0.00	0.71	194	205
SVM	0.00	0.40	198	207
ANN	0.00	1.21	173	187
Smina	0.93	2.36	2	164
RF-Score-VS	4.15	6.40	42	87
CNN-Score	0.39	2.28	91	185
Hard test set (438 PDB entries)				
RF	0.00	0.68	400	400
XGB	0.00	0.92	390	390
SVM	0.00	0.49	404	404
ANN	0.00	1.51	363	363
Smina	6.27	6.75	66	66
RF-Score-VS	6.19	6.38	124	124
CNN-Score	0.00	2.89	295	295

Strong docking power:

ClassyPose identified the native ligand pose as top-ranked solution in:

- ✓ 90% of the test entries
- ✓ 92.24% of the hard test entries

a)In Å. The value provided for each ML model is the average across five training-test runs. b)The number provided for each ML model is the median across five training-test runs.

V.K. Tran-Nguyen *et al.*, *Adv. Intell. Syst.* 2024, 6, 2400238

DOCKING POWER: POSE GENERATION & SELECTION

The emergence of end-to-end machine learning-based docking tools

DeepDock

- ⚙ Geometric deep learning for optimal protein–ligand binding (potential minimization)
- ✓ Target-specific features; interpretable learned potentials; strength in lead optimization
- ✗ Predominantly physically/chemically implausible docking poses; dependence on high-quality training complexes

KarmaDock

- ⚙ Graph neural networks and self-attention mechanisms; pose refinement based on protein–ligand and intramolecular interactions
- ✓ Scalability to ultra-large libraries
- ✗ Predominantly physically/chemically implausible docking poses; limited de novo pose generation capability

CarsiDock

- ⚙ Deep learning–guided docking approach; large-scale pretraining on millions of predicted complexes
- ✓ Generalization to unseen targets; low-data effectiveness; end-to-end docking and screening
- ✗ Generation of implausible docking poses; pretraining dataset biases that can affect predictions; heavy computational cost for retraining

Interformer

- ⚙ Interaction-aware model for docking and binding affinity prediction; graph-transformer framework with a pseudo-Huber loss function
- ✓ End-to-end docking and screening; superior performance over traditional graph neural networks; accurate protein-ligand interaction modeling
- ✗ Generation of a small fraction of implausible docking poses

SurfDock

- ⚙ Diffusion-based generative model operating on non-Euclidean manifolds; molecular surface feature–informed generation
- ✓ Generation of physically consistent poses; handling of apo structures and conformational variability
- ✗ Generation of implausible docking poses; requirement for accurate surface computation; computational cost of generative sampling

V.K. Tran-Nguyen and A.C. Camproux, *Curr. Opin. Struct. Biol.* 2025, 95, 103152

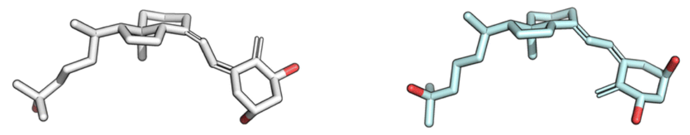
DOCKING POWER: POSE GENERATION & SELECTION

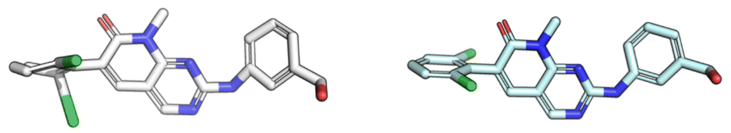
The importance of evaluating docking poses' physicochemical plausibility

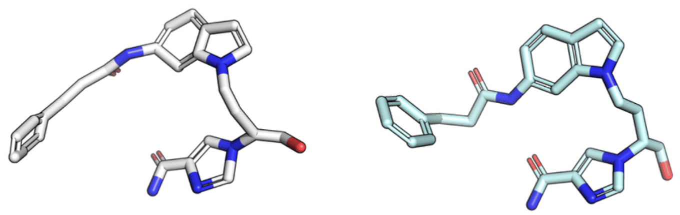
DOI: 10.1039/D3SC04185A (Edge Article) *Chem. Sci.*, 2024, **15**, 3130-3139

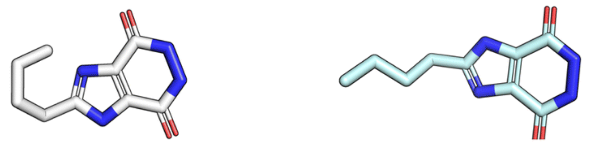
PoseBusters: AI-based docking methods fail to generate physically valid poses or generalise to novel sequences†

Martin Buttenschoen, Garrett M. Morris and Charlotte M. Deane *


(a) Double bond stereochemistry not preserved. DiffDock prediction for ligand VDX of protein-ligand complex 7QPP. RMSD 1.9 Å.


(b) Bond lengths too long. Uni-Mol prediction for ligand P16 of protein-ligand complex 1OPK. RMSD 1.5 Å.

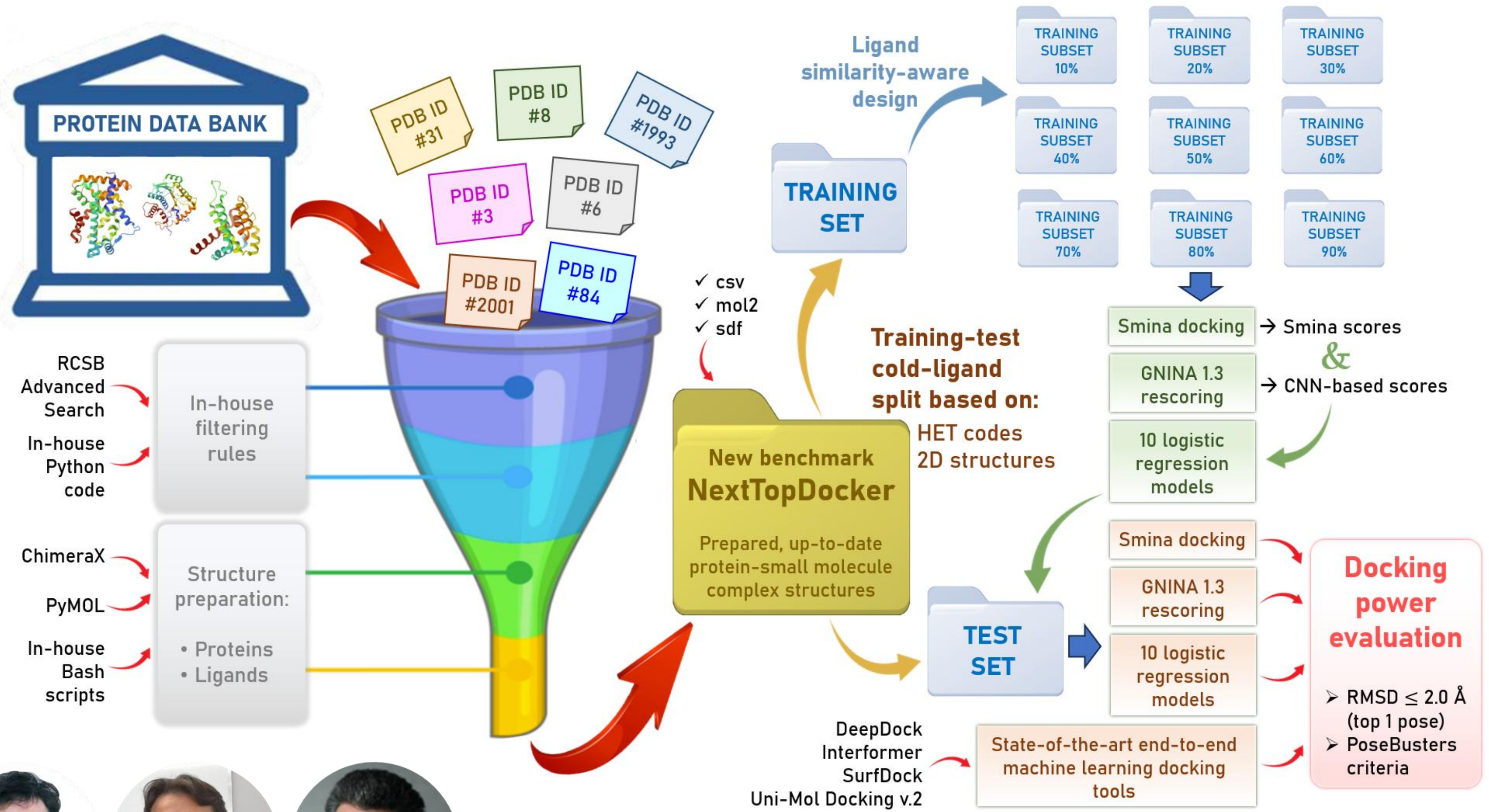

(c) Bond angles too extreme. Uni-Mol prediction for ligand FR4 of protein-ligand complex 1UML. RMSD 1.4 Å.


(d) Internal clash. DeepDock prediction for ligand BDI of protein-ligand complex 1N2V. RMSD 1.6 Å.

M. Buttenschoen *et al.*, *Chem. Sci.* 2024, 15, 3130

DOCKING POWER: POSE GENERATION & SELECTION

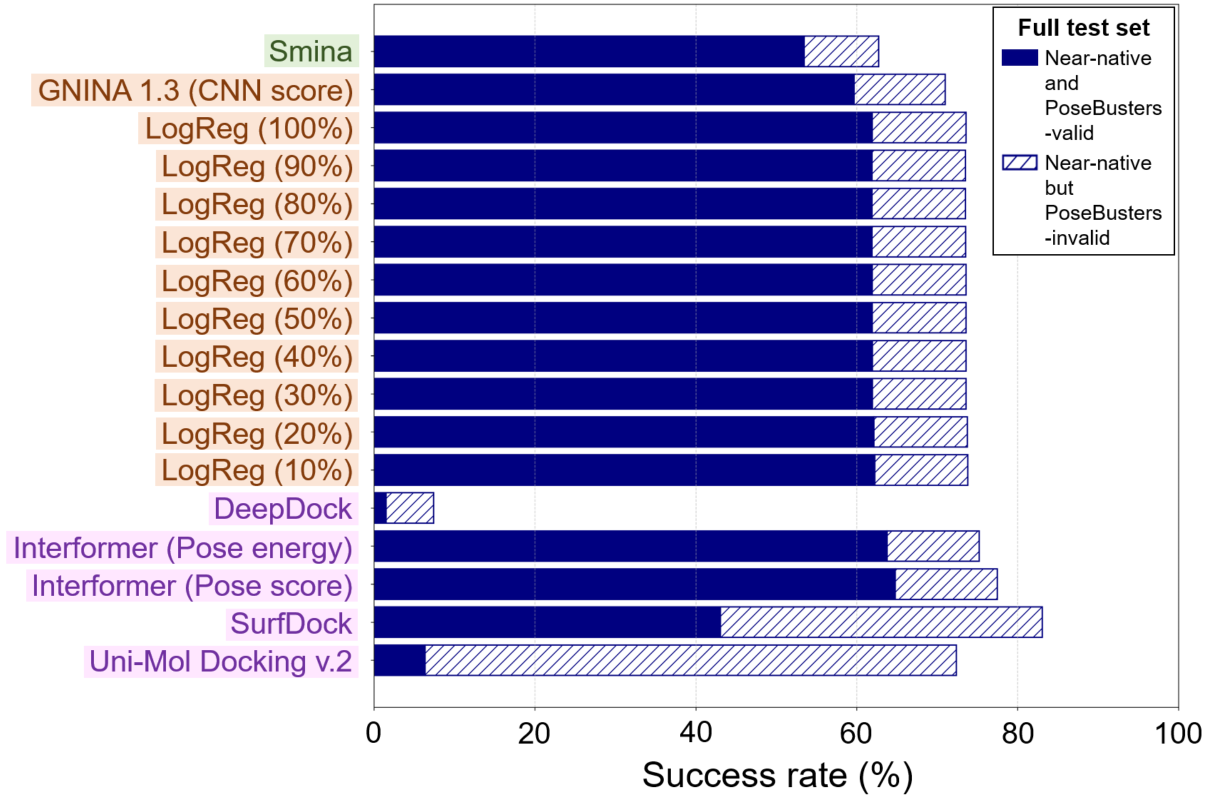
NextTopDocker: the largest-to-date docking power benchmark



C.M. Truong *et al.* (unpublished, under review)

DOCKING POWER: POSE GENERATION & SELECTION

NextTopDocker: the largest-to-date docking power benchmark



Smina	Fully physics-based pose sampling and pose scoring using Smina
GNINA 1.3 (CNN score) LogReg (x%)	Hybrid physics-based pose sampling (Smina) and ML-based rescoring of Smina-generated poses
DeepDock Interformer (Pose energy) Interformer (Pose score) SurfDock Uni-Mol Docking v.2	End-to-end ML docking tools: ML methodologies for both pose sampling and pose scoring

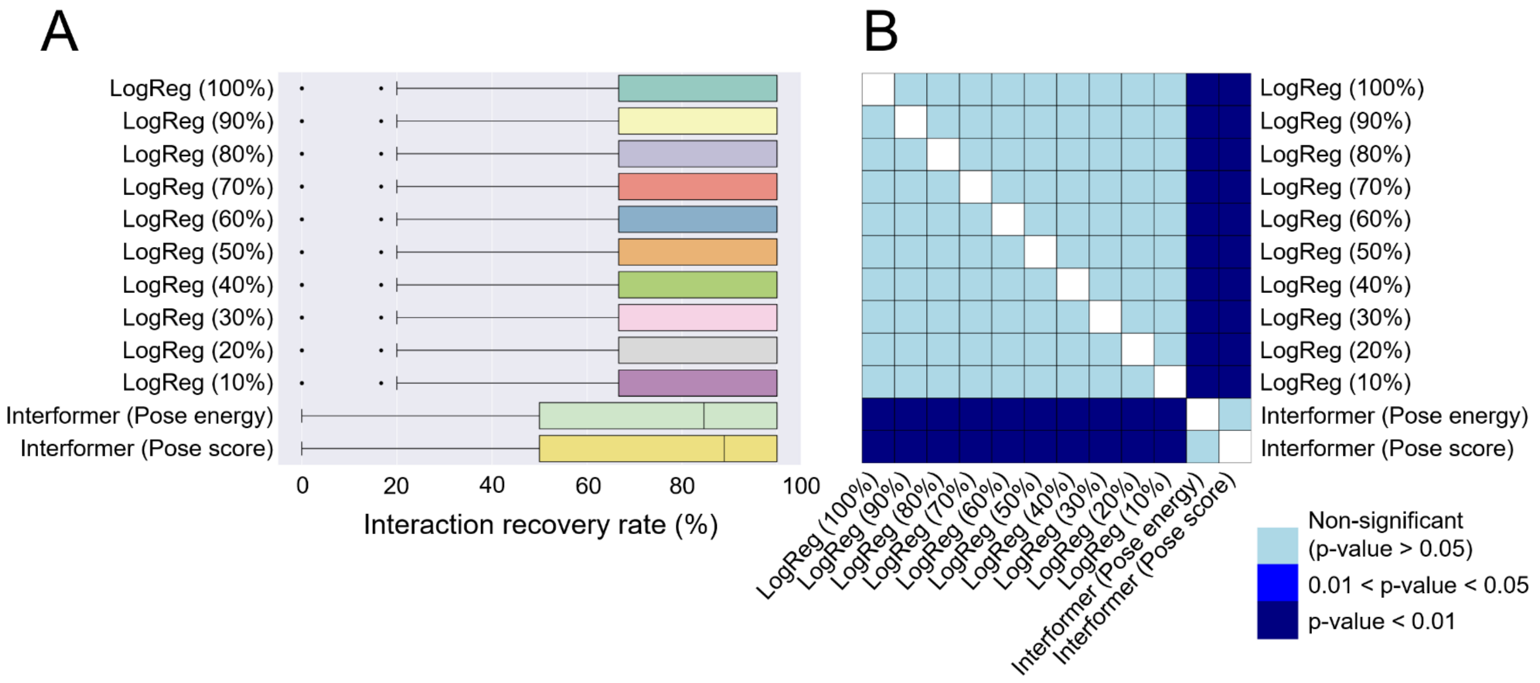
When physicochemical plausibility was evaluated, three of the four end-to-end ML docking tools failed to produce valid poses for >50% of the test set.

Only Interformer achieved performance comparable to our logistic regression-based rescoring models.

C.M. Truong *et al.* (unpublished, under review)

DOCKING POWER: POSE GENERATION & SELECTION

NextTopDocker: the largest-to-date docking power benchmark



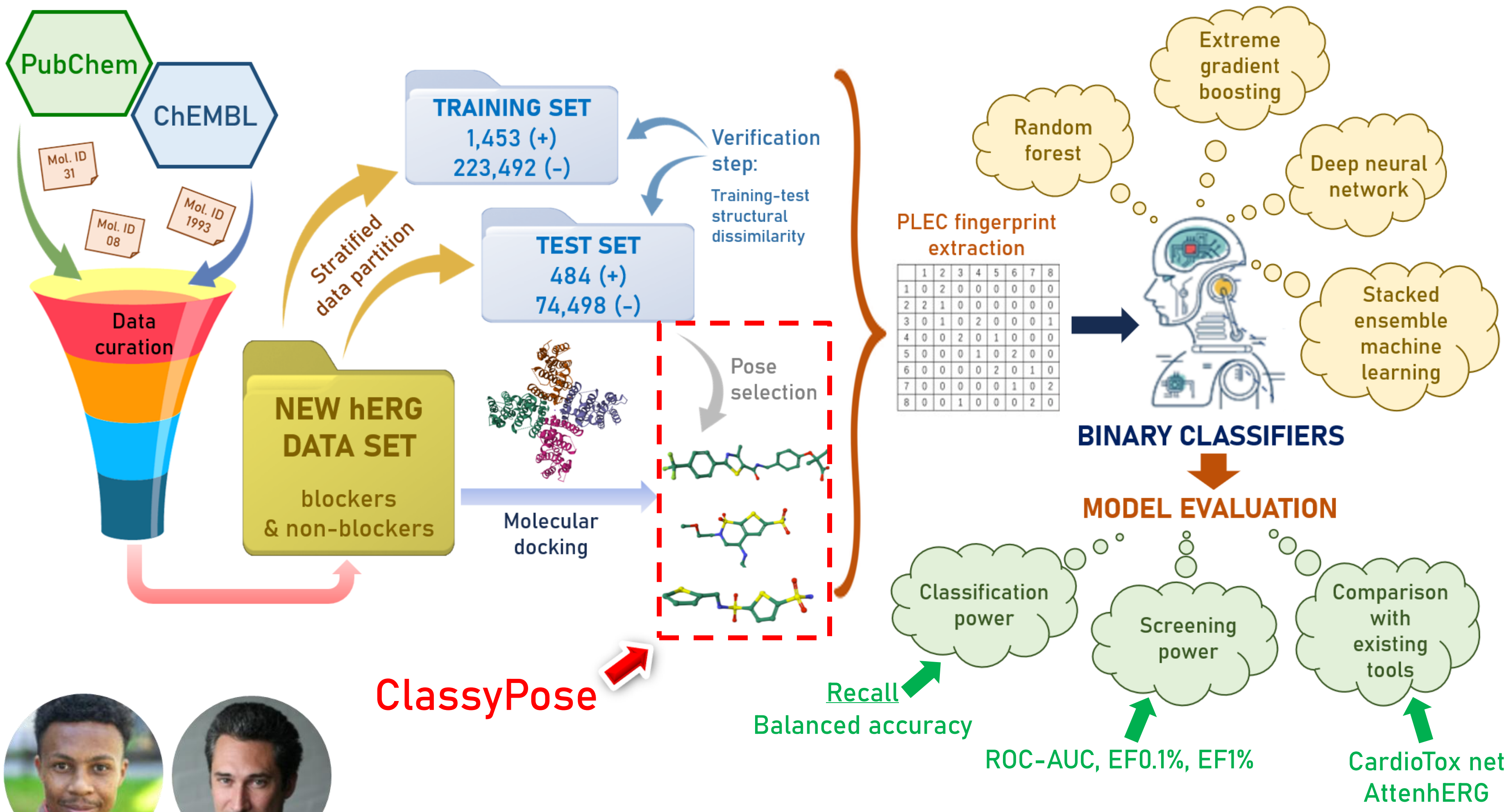
On the top-1 poses that were both near-native and PoseBusters-valid: all 10 logistic regression models recovered significantly more key non-covalent interactions (hydrogen bonds, ionic contacts, π-π stacking) than Interformer.

Robustness and generalization of hybrid physics-based pose generation followed by ML-based pose scoring over current end-to-end generative architectures

C.M. Truong *et al.* (unpublished, under review)

SCREENING POWER: ENRICHMENT FACTOR OF TRUE HITS

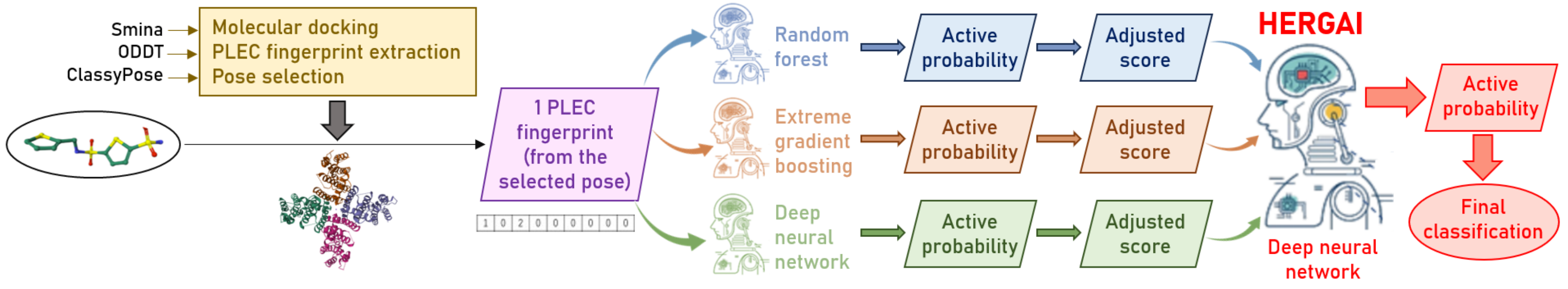
HERGAI: a machine-learning model for structure-based hERG inhibitor prediction

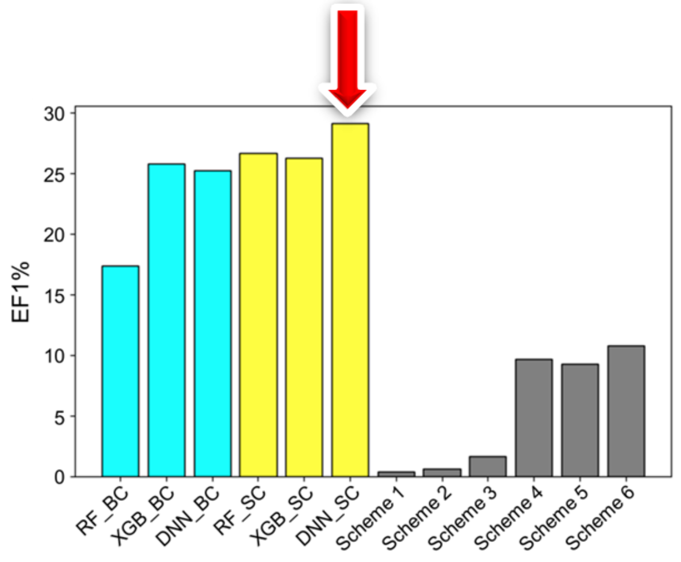


V.K. Tran-Nguyen *et al.*, *J. Cheminform.* 2025, 17, 110

SCREENING POWER: ENRICHMENT FACTOR OF TRUE HITS

HERGAI: a machine-learning model for structure-based hERG inhibitor prediction





	Classification performance		Screening performance	
	Recall	Balanced accuracy	Enrichment factor 0.1%	Enrichment factor 1.0%
HERGAI	0.86	0.76	64.05	29.13
CardioTox net	0.60	0.71	16.53	5.58
AttenhERG	0.70	0.72	51.65	16.74

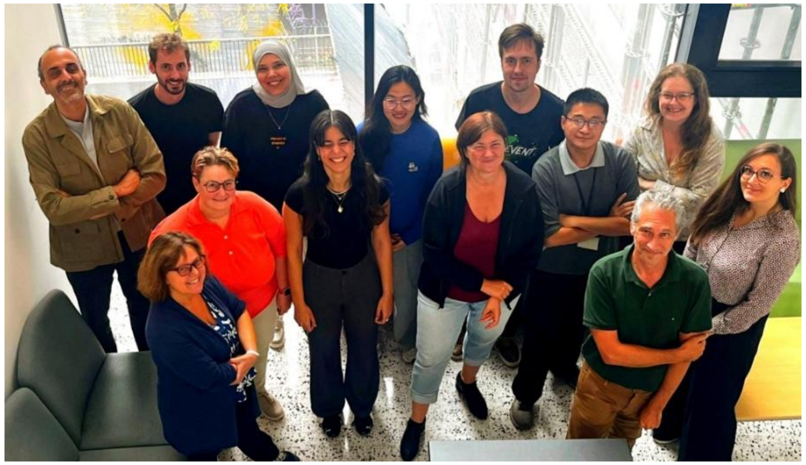
V.K. Tran-Nguyen *et al.*, *J. Cheminform.* 2025, 17, 110

CONCLUSION

- The primary bottleneck in molecular docking does not lie in pose generation, but rather in pose scoring and selection.
- Achieving more accurate and transferable docking predictions requires the integration of physically grounded principles.
- Physics-based docking algorithms, when combined with robust, interpretable machine-learning frameworks for pose scoring and selection, can deliver reliable and computationally efficient solutions to the docking problem, and contribute to enhancing screening performance.

ACKNOWLEDGMENT

My colleagues at BFA, UFR SDV, Université Paris Cité



Pr. Anne-Claude Camproux
Pr. Olivier Taboureau
Pr. Samuela Pasquali
Dr. Anne Badel
Dr. Delphine Flatters
Dr. Leslie Regad
Dr. Costas Bouyioukos
Mrs. Laurence Le Gall
Ms. Inés Sabine Rahali

My collaborator (outside Université Paris Cité)

Dr. Pedro J. Ballester (Imperial College London, UK)

My mentees at BFA, Université Paris Cité

Mr. Ulrick F. Randriharimanamizara (M2, 2024)
Mr. Cao-Minh Truong (M2, 2026)

My research funders

Université Paris Cité (Jan 2024-Dec 2024)
French National Research Agency (POLYPHAI project, JCJC, Nov 2025-Nov 2029)

INFORMATION FOR COLLABORATIONS

Email address: viet-khoa.tran-nguyen@u-paris.fr

Postal address:

Université Paris Cité, UFR Sciences du Vivant,

Unité de Biologie Fonctionnelle et Adaptative,

Bâtiment Lamarck A, 35 rue Hélène Brion, 75013 Paris

LinkedIn: https://www.linkedin.com/in/viet-khoa-tran-nguyen

ResearchGate: https://www.researchgate.net/profile/Viet-Khoa-Tran-Nguyen

ORCID: https://orcid.org/0000-0001-7497-333X

BACK-UP SLIDE

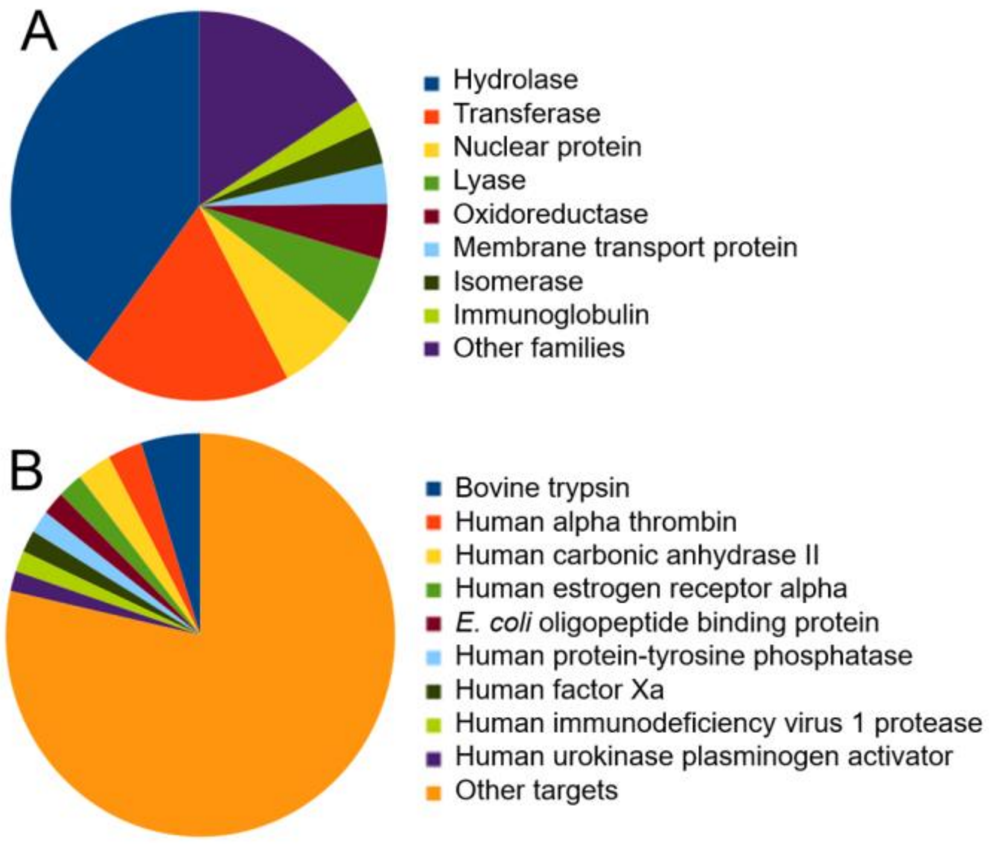
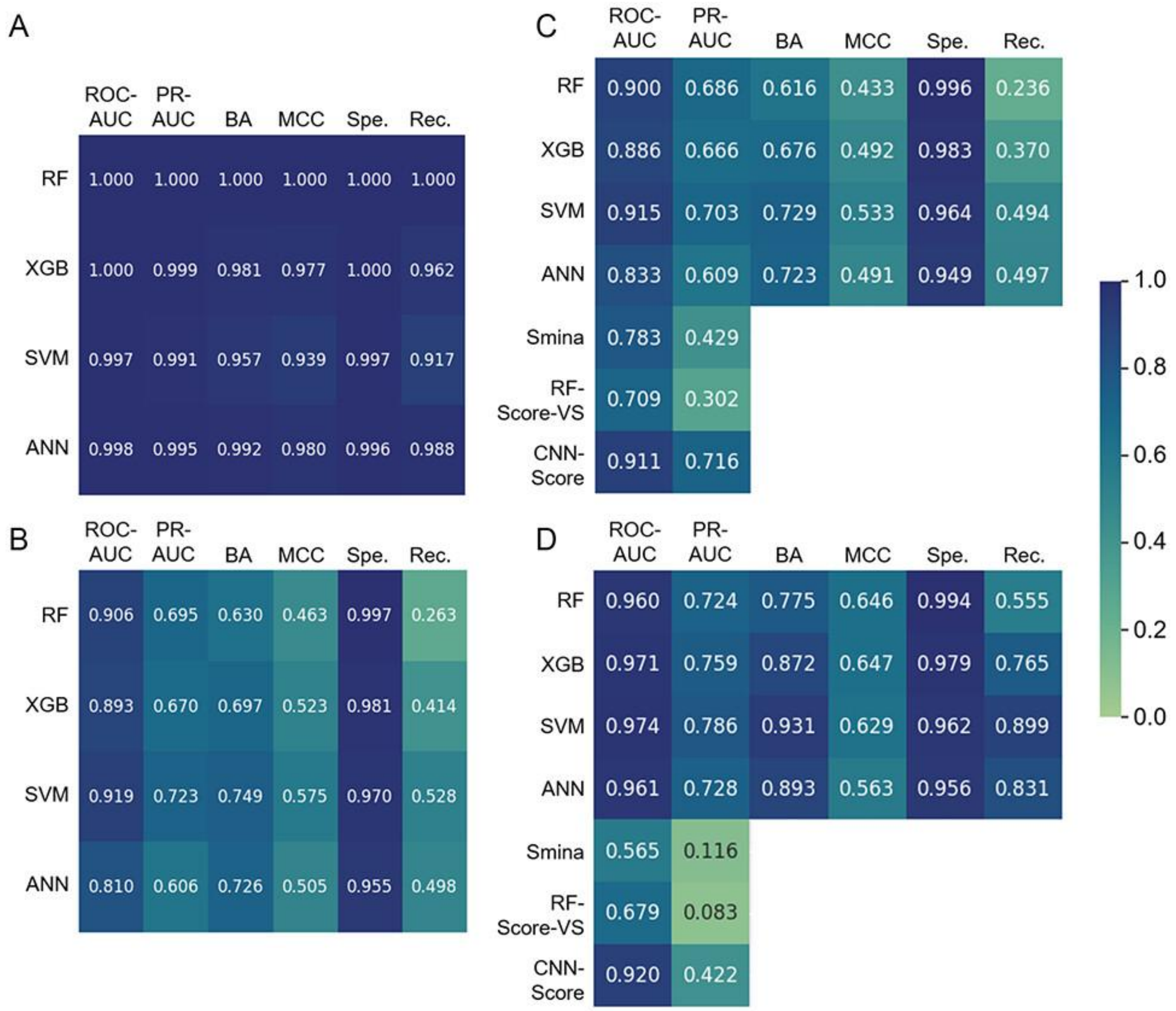


Figure S2. Target families having ≥ 35 representatives (A) and targets having ≥ 25 representatives (B). Large families frequently studied by biochemists also take up a large proportion of our retrieved PDB entries, e.g., hydrolases, transferases (including kinases), nuclear proteins (including nuclear G protein-coupled receptors), lyases (including hydro-lyases), membrane transport proteins (including ATP-binding cassette transporters). The protein targets having the most PDB representatives also belong to these families: bovine trypsin and human alpha thrombin (hydrolases), human carbonic anhydrase II (a hydro-lyase), or human estrogen receptor alpha (a nuclear receptor).

BACK-UP SLIDE



A

	ROC-AUC	PR-AUC	BA	MCC	Spe.	Rec.
RF	1.000	1.000	1.000	1.000	1.000	1.000
XGB	1.000	0.999	0.981	0.977	1.000	0.962
SVM	0.997	0.991	0.957	0.939	0.997	0.917
ANN	0.998	0.995	0.992	0.980	0.996	0.988

B

	ROC-AUC	PR-AUC	BA	MCC	Spe.	Rec.
RF	0.906	0.695	0.630	0.463	0.997	0.263
XGB	0.893	0.670	0.697	0.523	0.981	0.414
SVM	0.919	0.723	0.749	0.575	0.970	0.528
ANN	0.810	0.606	0.726	0.505	0.955	0.498

C

	ROC-AUC	PR-AUC	BA	MCC	Spe.	Rec.
RF	0.900	0.686	0.616	0.433	0.996	0.236
XGB	0.886	0.666	0.676	0.492	0.983	0.370
SVM	0.915	0.703	0.729	0.533	0.964	0.494
ANN	0.833	0.609	0.723	0.491	0.949	0.497
Smina	0.783	0.429				
RF-Score-VS	0.709	0.302				
CNN-Score	0.911	0.716				

D

	ROC-AUC	PR-AUC	BA	MCC	Spe.	Rec.
RF	0.960	0.724	0.775	0.646	0.994	0.555
XGB	0.971	0.759	0.872	0.647	0.979	0.765
SVM	0.974	0.786	0.931	0.629	0.962	0.899
ANN	0.961	0.728	0.893	0.563	0.956	0.831
Smina	0.565	0.116				
RF-Score-VS	0.679	0.083				
CNN-Score	0.920	0.422				

Heatmaps illustrating the performance of our ML classifiers in four pose classification scenarios:
(A) on the entire training set with which model training had been carried out;
(B) during 5-fold CV;
(C) on the test set;
(D) on the hard test set.

For each ML algorithm, 5 training-test runs were carried out per training-test partition, after which the average value across five runs of each metric was calculated and indicated.

V.K. Tran-Nguyen *et al.*, *Adv. Intell. Syst.* 2024, 6, 2400238

BACK-UP SLIDE

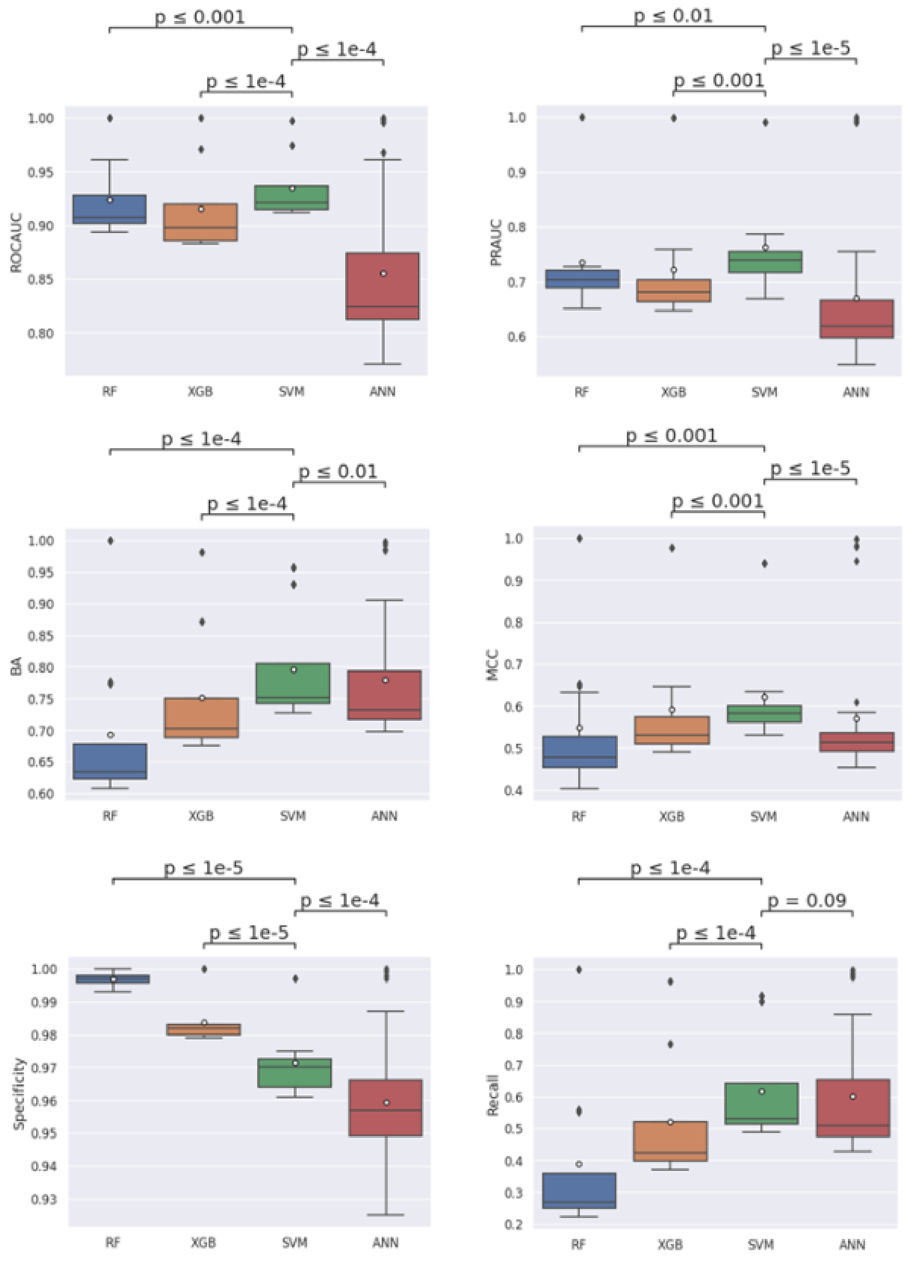
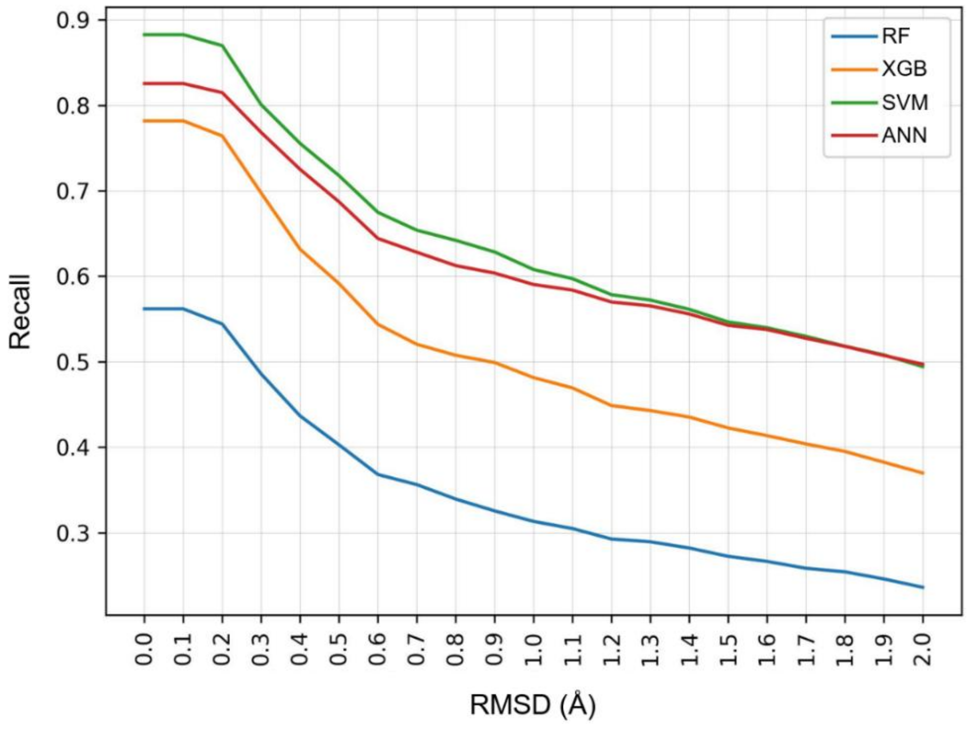


Figure S5. Comparison of ROC-AUC, PR-AUC, balanced accuracy (BA), Matthews correlation coefficient (MCC), specificity and recall values given by our four ML models in four pose classification scenarios seen in Figure 4 (five runs per model). Statistical analyses were performed using the Mann-Whitney-Wilcoxon test with Bonferroni correction. SVM gave significantly better performance than the other three ML classifiers (p-values < 0.05) in terms of ROC-AUC, PRAUC, BA, MCC, and recall. Only in terms of specificity did SVM give slightly lower performance than RF and XGB. Nevertheless, this model still gave near-perfect specificity values (> 0.96 in all cases).

BACK-UP SLIDE



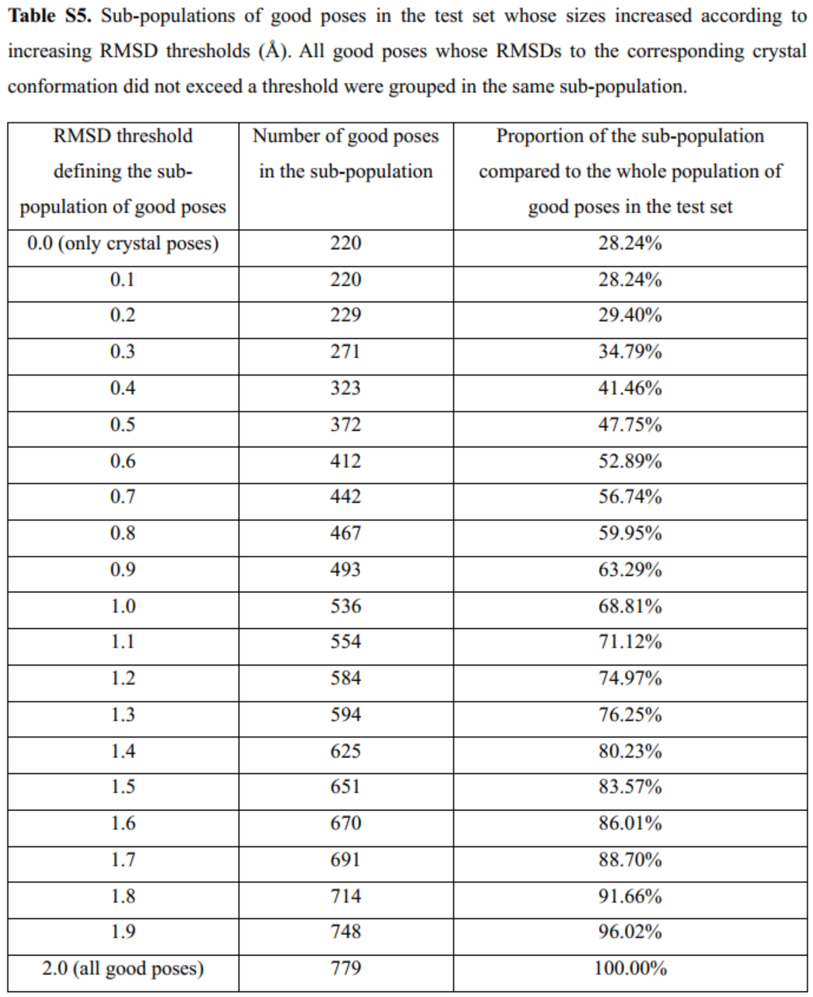


Table S5. Sub-populations of good poses in the test set whose sizes increased according to increasing RMSD thresholds (Å). All good poses whose RMSDs to the corresponding crystal conformation did not exceed a threshold were grouped in the same sub-population.

RMSD threshold defining the sub-population of good poses	Number of good poses in the sub-population	Proportion of the sub-population compared to the whole population of good poses in the test set
0.0 (only crystal poses)	220	28.24%
0.1	220	28.24%
0.2	229	29.40%
0.3	271	34.79%
0.4	323	41.46%
0.5	372	47.75%
0.6	412	52.89%
0.7	442	56.74%
0.8	467	59.95%
0.9	493	63.29%
1.0	536	68.81%
1.1	554	71.12%
1.2	584	74.97%
1.3	594	76.25%
1.4	625	80.23%
1.5	651	83.57%
1.6	670	86.01%
1.7	691	88.70%
1.8	714	91.66%
1.9	748	96.02%
2.0 (all good poses)	779	100.00%

Recall performance of our ML algorithms on sub-populations of good poses in the test set whose sizes increased according to increasing RMSD thresholds (Å).
The average recall value across five training-test runs was computed for each algorithm per good pose sub-population.

V.K. Tran-Nguyen *et al.*, *Adv. Intell. Syst.* 2024, 6, 2400238

BACK-UP SLIDE

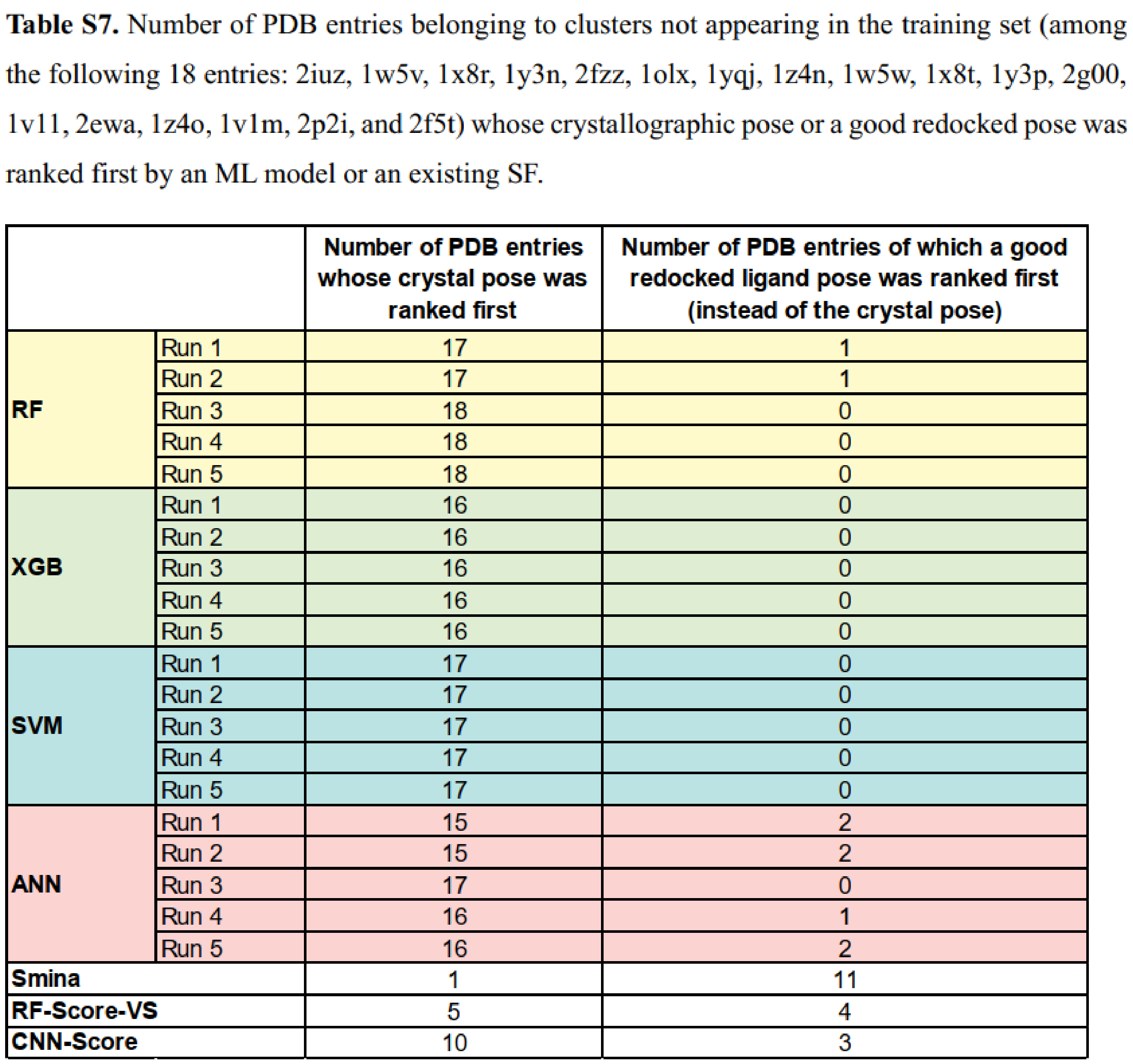


Table S7. Number of PDB entries belonging to clusters not appearing in the training set (among the following 18 entries: 2iuz, 1w5v, 1x8r, 1y3n, 2fzz, 1olx, 1yqj, 1z4n, 1w5w, 1x8t, 1y3p, 2g00, 1v11, 2ewa, 1z4o, 1v1m, 2p2i, and 2f5t) whose crystallographic pose or a good redocked pose was ranked first by an ML model or an existing SF.

		Number of PDB entries whose crystal pose was ranked first	**Number of PDB entries of which a good redocked ligand pose was ranked first (instead of the crystal pose)**
RF	Run 1	17	1
	Run 2	17	1
	Run 3	18	0
	Run 4	18	0
	Run 5	18	0
XGB	Run 1	16	0
	Run 2	16	0
	Run 3	16	0
	Run 4	16	0
	Run 5	16	0
SVM	Run 1	17	0
	Run 2	17	0
	Run 3	17	0
	Run 4	17	0
	Run 5	17	0
ANN	Run 1	15	2
	Run 2	15	2
	Run 3	17	0
	Run 4	16	1
	Run 5	16	2
Smina		1	11
RF-Score-VS		5	4
CNN-Score		10	3

BACK-UP SLIDE

Table S8. Percentage of actives and inactives in the top 1%-ranked population of a VS scheme involving pose selection and ligand ranking by the same existing SF that remained in the top 1% when pose selection was carried out by our SVM model (good pose probability) instead.

Data set	SDS-SDS → SVMGPP-SDS[1]		RFSVS-RFSVS → SVMGPP-RFSVS[2]		CNNS-CNNS → SVMGPP-CNNS[3]	
	Actives	Inactives	Actives	Inactives	Actives	Inactives
PPARA_OTS	100%	70.37%	100%	90.91%	50.00%	36.36%
HXK4_DUD-E	100%	57.89%	83.33%	70.73%	50.00%	37.21%
ROCK1_DUD-E	100%	67.69%	81.58%	48.00%	88.89%	53.70%
EGFR_DEKOIS2.0	N/A[*]	76.92%	100%	60.00%	100%	72.73%
11BHSD1_DEKOIS2.0	100%	58.33%	100%	55.56%	100%	27.27%

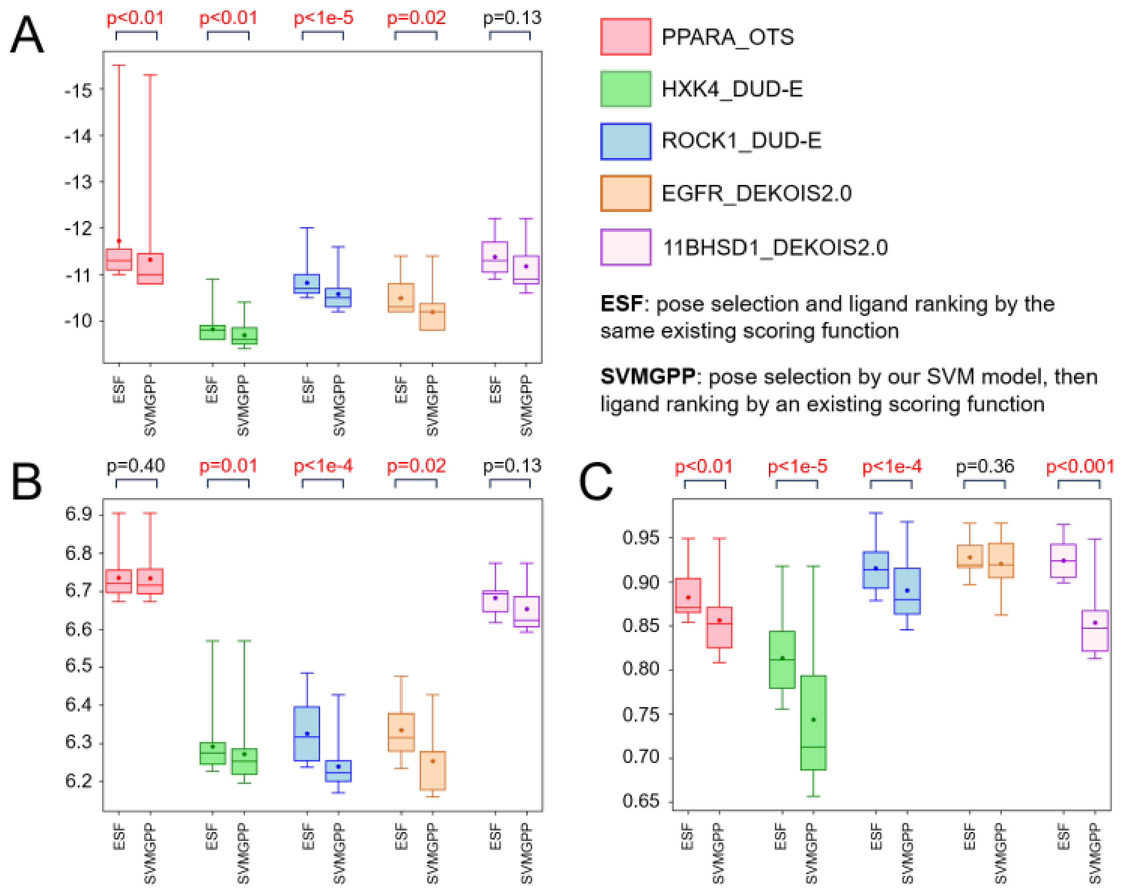
[1] Ligand ranking by Smina docking scores after pose selection by our SVM model instead of Smina docking scores.

[2] Ligand ranking by RF-Score-VS after pose selection by our SVM model instead of RF-Score-VS.

[3] Ligand ranking by CNN-Score after pose selection by our SVM model instead of CNN-Score.

[*] Pose selection and ligand ranking by Smina docking scores did not retrieve any true actives in the top 1%-ranked population (EF1% = 0.00).

BACK-UP SLIDE



ESF: pose selection and ligand ranking by the same existing scoring function

SVMGPP: pose selection by our SVM model, then ligand ranking by an existing scoring function

Figure S6. Score distributions of the inactive ligands in the top 1%-ranked population given by: (A) Smina, (B) RF-Score-VS, and (C) CNN-Score. It is observed that the scores of top-ranked inactives given by an existing SF generally worsened when pose selection was carried out by our SVM model (good pose probability), in comparison to when the same existing SF was used for both pose selection and ligand ranking: statistical analyses were performed using the Mann-Whitney-Wilcoxon test, most p-values were < 0.05 (marked in red), indicating that most differences were statistically significant.

BACK-UP SLIDE

Table 1. Final composition of our training set and test set.

	Activity label	Training set	Test set
Number of BM scaffolds	Active	796	252
	Inactive	61,464	21,445
Number of ligands	Active	1,453	484
	Inactive	223,492	74,498

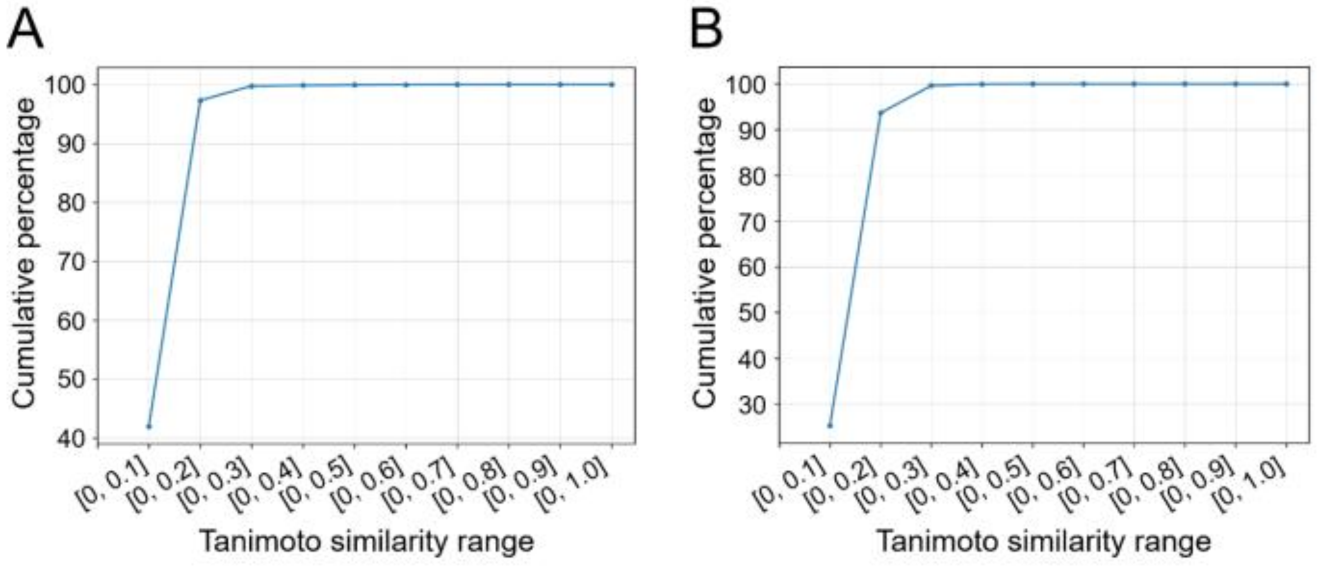
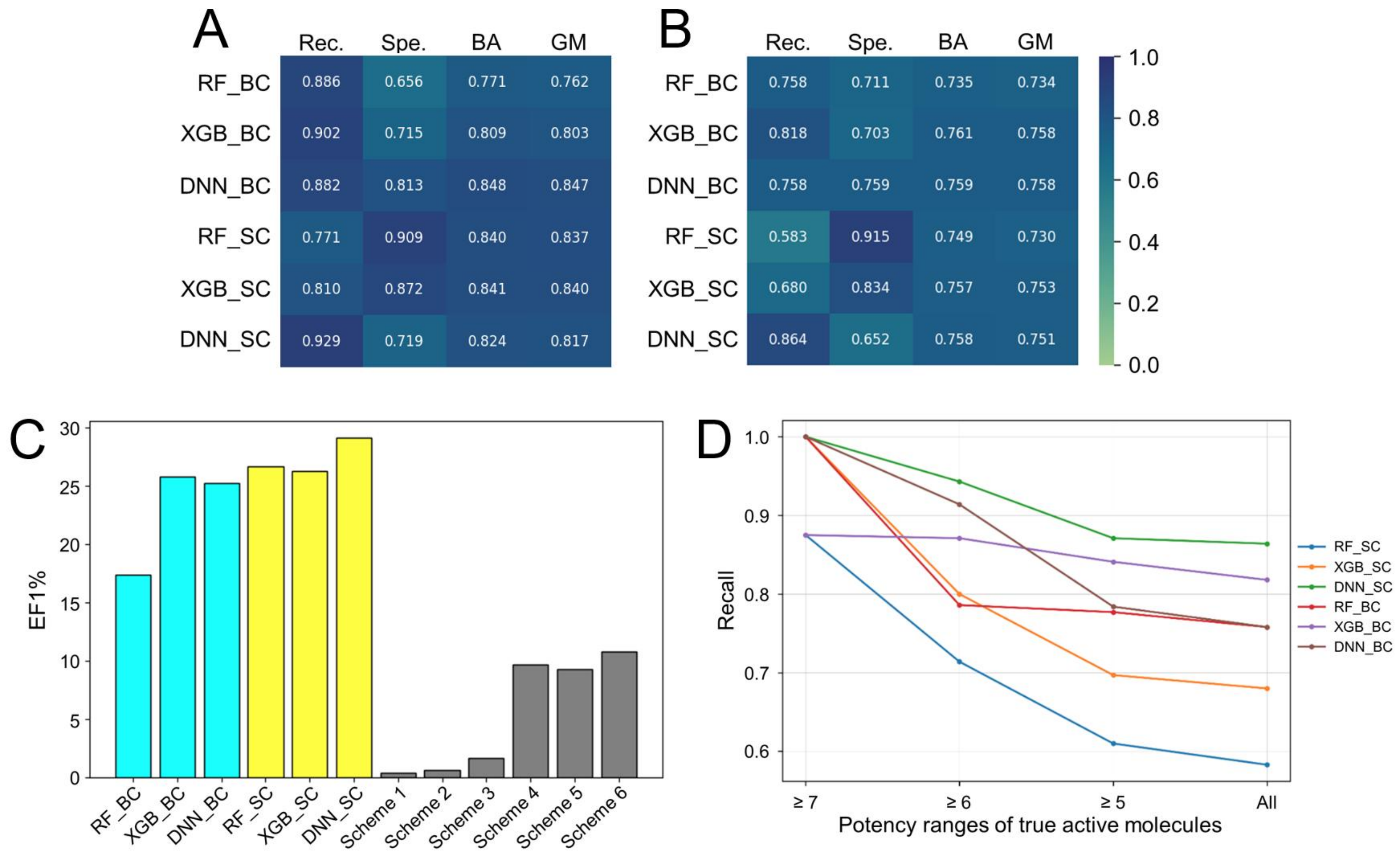


Figure 2. Cumulative percentage of training active-test active pairs **(A)** and training inactive-test inactive pairs **(B)** whose Tanimoto similarity coefficients fell within defined ranges, in terms of Morgan fingerprints (2,048 bits, radius 2).

BACK-UP SLIDE



A

	Rec.	Spe.	BA	GM
RF_BC	0.886	0.656	0.771	0.762
XGB_BC	0.902	0.715	0.809	0.803
DNN_BC	0.882	0.813	0.848	0.847
RF_SC	0.771	0.909	0.840	0.837
XGB_SC	0.810	0.872	0.841	0.840
DNN_SC	0.929	0.719	0.824	0.817

B

	Rec.	Spe.	BA	GM
RF_BC	0.758	0.711	0.735	0.734
XGB_BC	0.818	0.703	0.761	0.758
DNN_BC	0.758	0.759	0.759	0.758
RF_SC	0.583	0.915	0.749	0.730
XGB_SC	0.680	0.834	0.757	0.753
DNN_SC	0.864	0.652	0.758	0.751

BACK-UP SLIDE

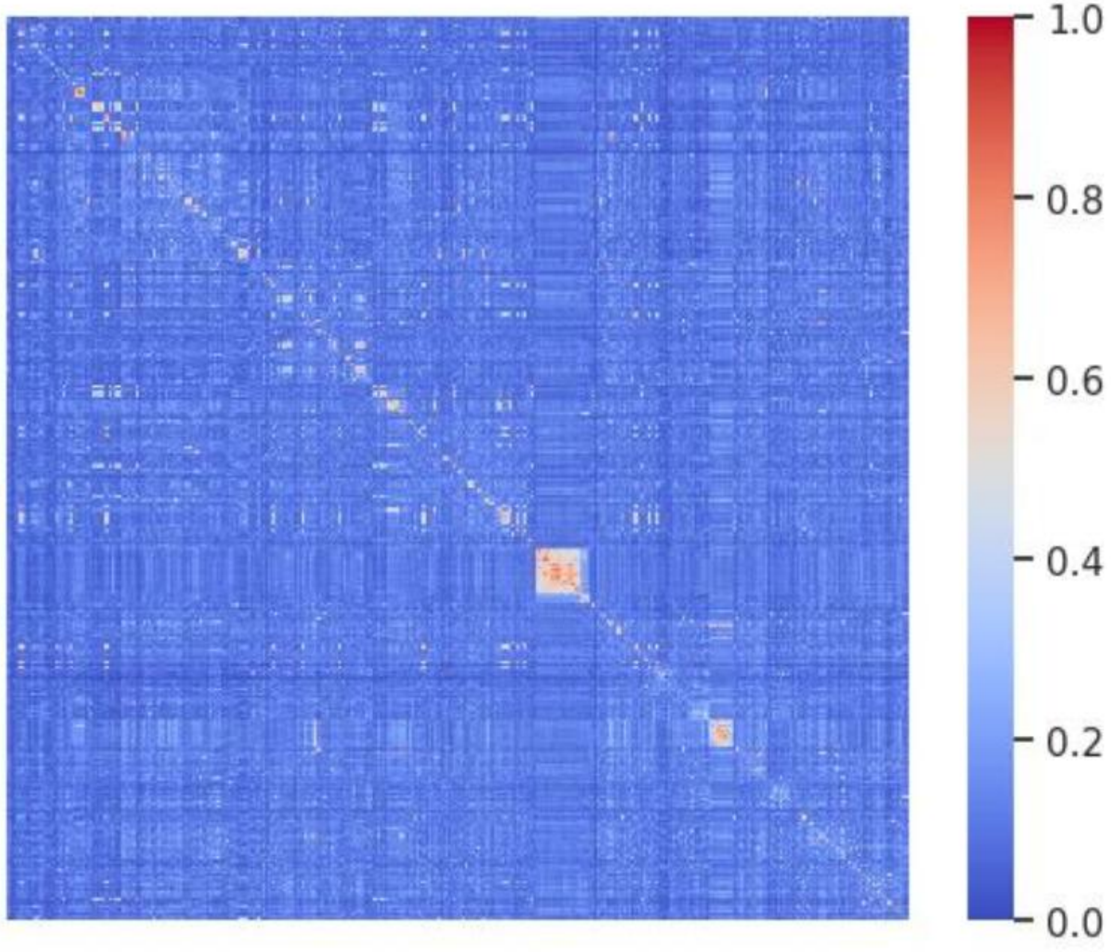


Figure S1. Heatmap illustrating the pairwise Tanimoto similarity values of 1,937 true active molecules included in our final data set, in terms of Morgan fingerprints (2,048 bits, radius 2). Only 1.50% of all active ligand pairs (n=1,875,016) had Tanimoto similarity above 0.30, highlighting the structural diversity of our active ligands.

BACK-UP SLIDE

Table S4. The eight strongest hERG blockers in our test set (IC_{50}s not exceeding 0.1 μM) and their most similar active ligands used in model training, in terms of Morgan fingerprints (2,048 bits, radius 2). The Tanimoto similarity scores are provided, proving the structural difference between them.

Strongest hERG blocker in the test set (SID)	Most similar active ligand in the training set (SID)	Tanimoto similarity
103626969	103528194	0.305
377003148	103609934	0.281
377003149	404958464	0.299
377003131	404958431	0.293
377003143	312363999	0.333
377003150	103609934	0.292
377003146	123089189	0.311
336951486	442142298	0.533

BACK-UP SLIDE

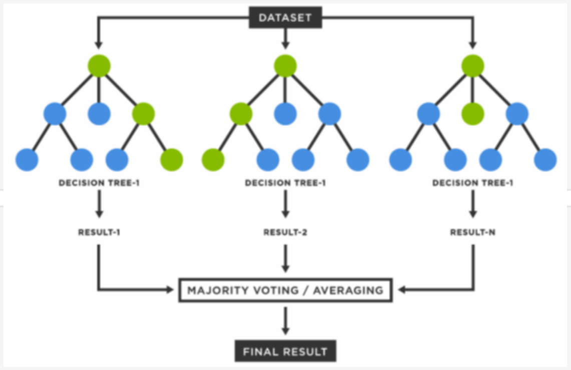
Random Forest

- An ensemble of parallel models working on the bagging principle.
- Each of these models is an individual decision tree (DT) generated and independently trained on a different sample taken randomly (with replacement) from the same training population (bootstrap).
- All models would then issue the requested output for new data by means of majority voting or averaging individual predictions (aggregation).
- By using different samples selected from the training data in an aleatory manner, an RF model avoids overfitting a single training set, and has been shown to outperform individual DTs.
- We use the RandomForestClassifier function from the sklearn Python package.

```
#Train the RF model on the training molecules:
rf_plec = RandomForestClassifier(n_estimators = 400, max_features = 'sqrt', n_jobs = 30)
rf_plec.fit(train_features, Train_Class)

#Test the RF model on the test molecules:
prediction_test_rf_plec_class = rf_plec.predict(test_features)
prediction_test_rf_plec_prob = rf_plec.predict_proba(test_features)
```

```
#Get virtual screening results on the test molecules and export the resulting hit list to a csv file:
plec_result_rf = pd.DataFrame({"Active_Prob": prediction_test_rf_plec_prob[:, 0],
                               "Inactive_Prob": prediction_test_rf_plec_prob[:, 1],
                               "Predicted_Class": prediction_test_rf_plec_class,
                               "Real_Class": Test_Class})
plec_result_rf.to_csv("Provide_the_pathway_to_a_directory_to_store_your_csv_hit_list")
```



BACK-UP SLIDE

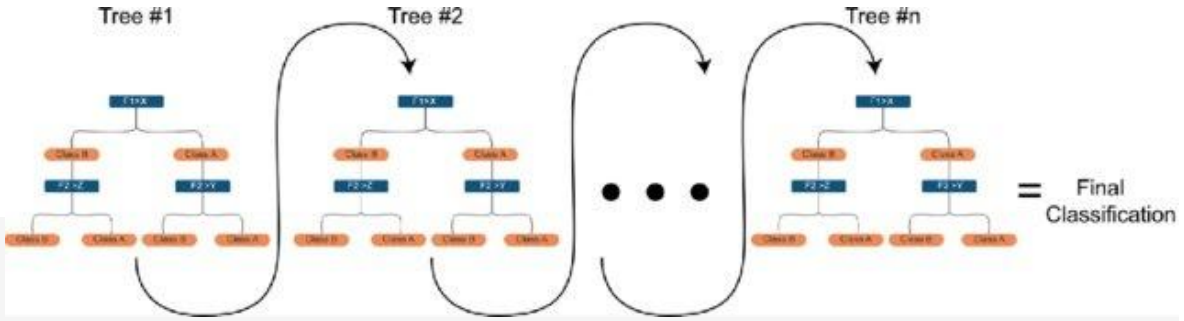
Random Forest

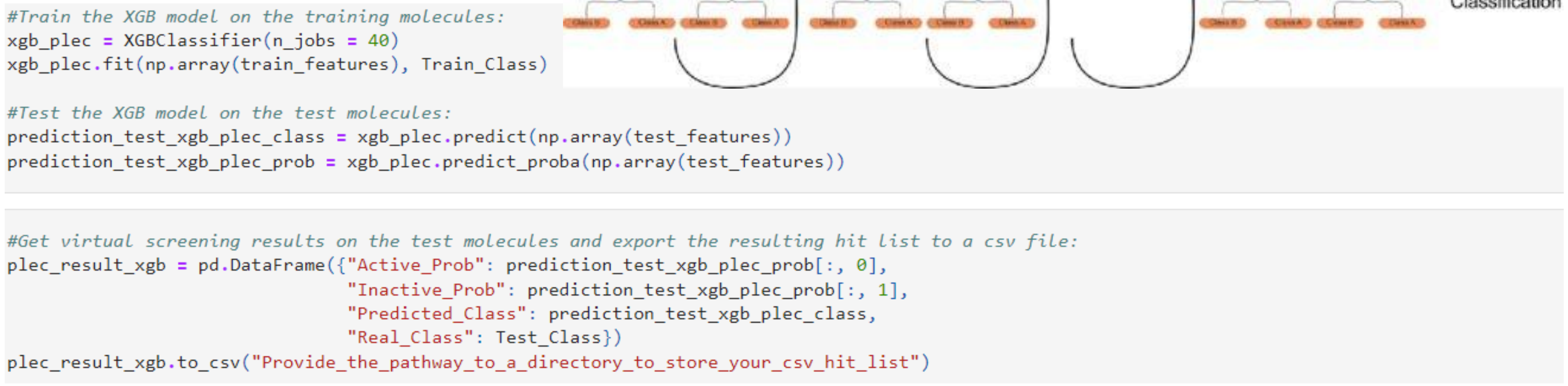
- **Advantages:**
  - ✓ It tends not to overfit. No single tree sees all the data. This helps to focus on the general patterns within the training data and reduce sensitivity to noise.
  - ✓ Accuracy calculated from out-of-bag samples is a proxy for using a separate test data set. The out-of-bag samples are those not used for training a specific tree and as such can be used as an unbiased measure of performance.
  - ✓ It is flexible to both classification and regression problems.
  - ✓ It works well with both categorical and continuous values.
- **Disadvantages:**
  - ✓ It requires much computational power as well as resources as it builds numerous trees to combine their outputs: a trained forest may require significant memory for storage, due to the need for retaining the information from several hundred individual trees.
  - ✓ It also requires much time for training as it combines a lot of DTs to determine the class.

BACK-UP SLIDE

Extreme Gradient Boosting

- An ensemble of sequential models working on the boosting principle.
- It builds DTs consecutively and tries to learn from wrongly classified observations by adding a higher weight on them in the subsequently built DTs.
- Prediction errors of early models are minimized as later ones are added, meaning the final model is the most accurate and the best-performing.
- We use the XGBClassifier function from the xgboost Python library to build XGB models.





```
#Train the XGB model on the training molecules:
xgb_plec = XGBClassifier(n_jobs = 40)
xgb_plec.fit(np.array(train_features), Train_Class)

#Test the XGB model on the test molecules:
prediction_test_xgb_plec_class = xgb_plec.predict(np.array(test_features))
prediction_test_xgb_plec_prob = xgb_plec.predict_proba(np.array(test_features))

#Get virtual screening results on the test molecules and export the resulting hit list to a csv file:
plec_result_xgb = pd.DataFrame({"Active_Prob": prediction_test_xgb_plec_prob[:, 0],
                                "Inactive_Prob": prediction_test_xgb_plec_prob[:, 1],
                                "Predicted_Class": prediction_test_xgb_plec_class,
                                "Real_Class": Test_Class})
plec_result_xgb.to_csv("Provide_the_pathway_to_a_directory_to_store_your_csv_hit_list")
```

BACK-UP SLIDE

Extreme Gradient Boosting

- **Advantages:**
  - ✓ It is an easy-to-read and interpret algorithm, making its prediction interpretations easy to handle.
  - ✓ The prediction capability is efficient.
- **Disadvantages:**
  - ✓ XGB is very sensitive to outliers since every classifier is forced to fix the errors in predecessor learners → can overemphasize outliers and cause overfitting.
  - ✓ Computationally expensive - often require many trees (>1000) which can be time and memory exhaustive.

BACK-UP SLIDE

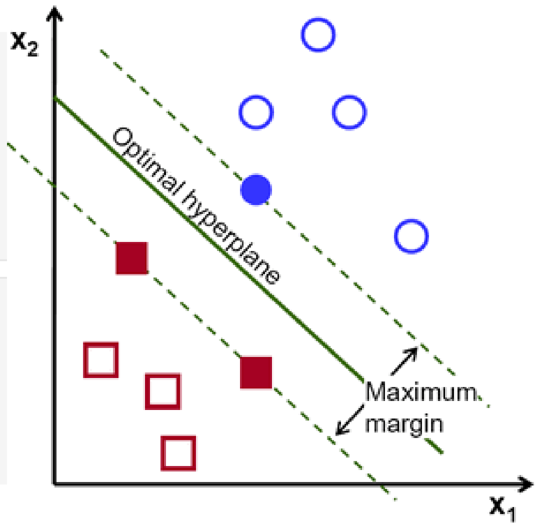
Support Vector Machine

- It views each data point (instance) as an *n*-dimensional vector, and constructs an (*n*-1)-dimensional hyperplane that separates those points such that the distance from it to the nearest data point is maximized.
- In case the data points cannot be linearly separated in the given *n*-dimensional space, a kernel function, e.g. the radial basis function (RBF) kernel, is used to map such data into a much higher-dimensional space where they are linearly separable.
- An SVM model can do both linear and non-linear analyses.
- The SVC function from the sklearn Python library is used.

```python
#Train the SVM model on the training molecules:
svm_plec = SVC(degree = 3, kernel = "rbf", probability = True)
svm_plec.fit(train_features, Train_Class)

#Test the SVM model on the test molecules:
prediction_test_svm_plec_class = svm_plec.predict(test_features)
prediction_test_svm_plec_prob = svm_plec.predict_proba(test_features)
```

```python
#Get virtual screening results on the test molecules and export the resulting hit list to a csv file:
plec_result_svm  = pd.DataFrame({"Active_Prob": prediction_test_svm_plec_prob[:, 0],
                                "Inactive_Prob": prediction_test_svm_plec_prob[:, 1],
                                "Predicted_Class": prediction_test_svm_plec_class,
                                "Real_Class": Test_Class})
plec_result_svm.to_csv("Provide_the_pathway_to_a_directory_to_store_your_csv_hit_list")
```



BACK-UP SLIDE

Support Vector Machine

- **Advantages:**
  - ✓ SVM is very effective even with high dimensional data.
  - ✓ SVM can perform well on a data set where the number of features is higher than the number of rows of data.
  - ✓ SVM works very well when the classes are well separated.
  - ✓ SVM can be used for both regression and classification problems.
  - ✓ SVM can work well with image data.
- **Disadvantages:**
  - ✓ If the classes are not well separated (there are overlapping classes), SVM does not perform well.
  - ✓ Choosing a “good” kernel function is not easy.
  - ✓ Training time may be long for large data sets.
  - ✓ It is difficult to understand and interpret an SVM model (compared to DTs).

BACK-UP SLIDE

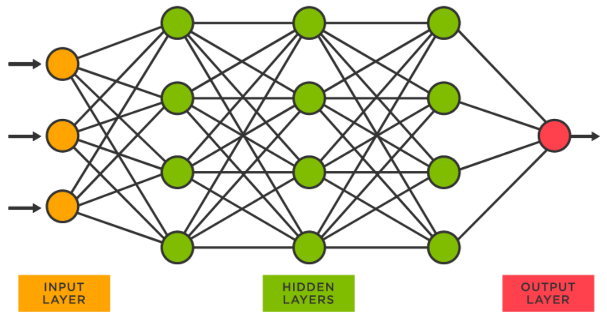
Artificial Neural Network

- A supervised learning system taking inspiration from the human brain.
- An ANN is generally comprised of 01 input layer, ≥ 01 hidden layer(s) and 01 output layer.
- The data, after being fed to the input nodes, are passed to the hidden nodes, which are constituted by multiple neurons where all the computations are performed.
- An ANN model can do both linear and non-linear analyses.
- An ANN is prone to overfitting the training instances (large number of data descriptors and their coefficients processed by the hidden nodes) → large weights assigned to certain descriptors can be penalized by the "weight decay" method, which regularizes the model.
- We use the MLPClassifier function from the sklearn Python library to build ANN models.

```
#Train the ANN model on the training molecules:
ann_plec = MLPClassifier(max_iter = 9000)
ann_plec.fit(train_features, Train_Class)

#Test the ANN model on the test molecules:
prediction_test_ann_plec_class = ann_plec.predict(test_features)
prediction_test_ann_plec_prob = ann_plec.predict_proba(test_features)
```

```
#Get virtual screening results on the test molecules and export the resulting hit list to a csv file:
plec_result_ann = pd.DataFrame({"Active_Prob": prediction_test_ann_plec_prob[:, 0],
                                "Inactive_Prob": prediction_test_ann_plec_prob[:, 1],
                                "Predicted_Class": prediction_test_ann_plec_class,
                                "Real_Class": Test_Class})
plec_result_ann.to_csv("Provide_the_pathway_to_a_directory_to_store_your_csv_hit_list")
```



BACK-UP SLIDE

Artificial Neural Network

- **Advantages:**
  - ✓ ANNs are capable of performing more complex tasks and activities as compared to other machines (e.g. analyzing visual information and segregating it into different categories).
  - ✓ By processing, segregating, and categorizing unorganized data, ANNs can very well organize data.
  - ✓ Their structure is adaptive in nature: for whatever purpose an ANN is applied, it alters its course of the structure according to the purpose.
- **Disadvantages:**
  - ✓ ANNs require heavy machinery and hardware equipment to work for any application.
  - ✓ ANNs are highly dependent on the data made available to them. The efficiency of any ANN is directly proportional to the amount of data it receives to process.
  - ✓ Trainers have minimal control over the actual performance and overall functioning of ANNs.

BACK-UP SLIDE

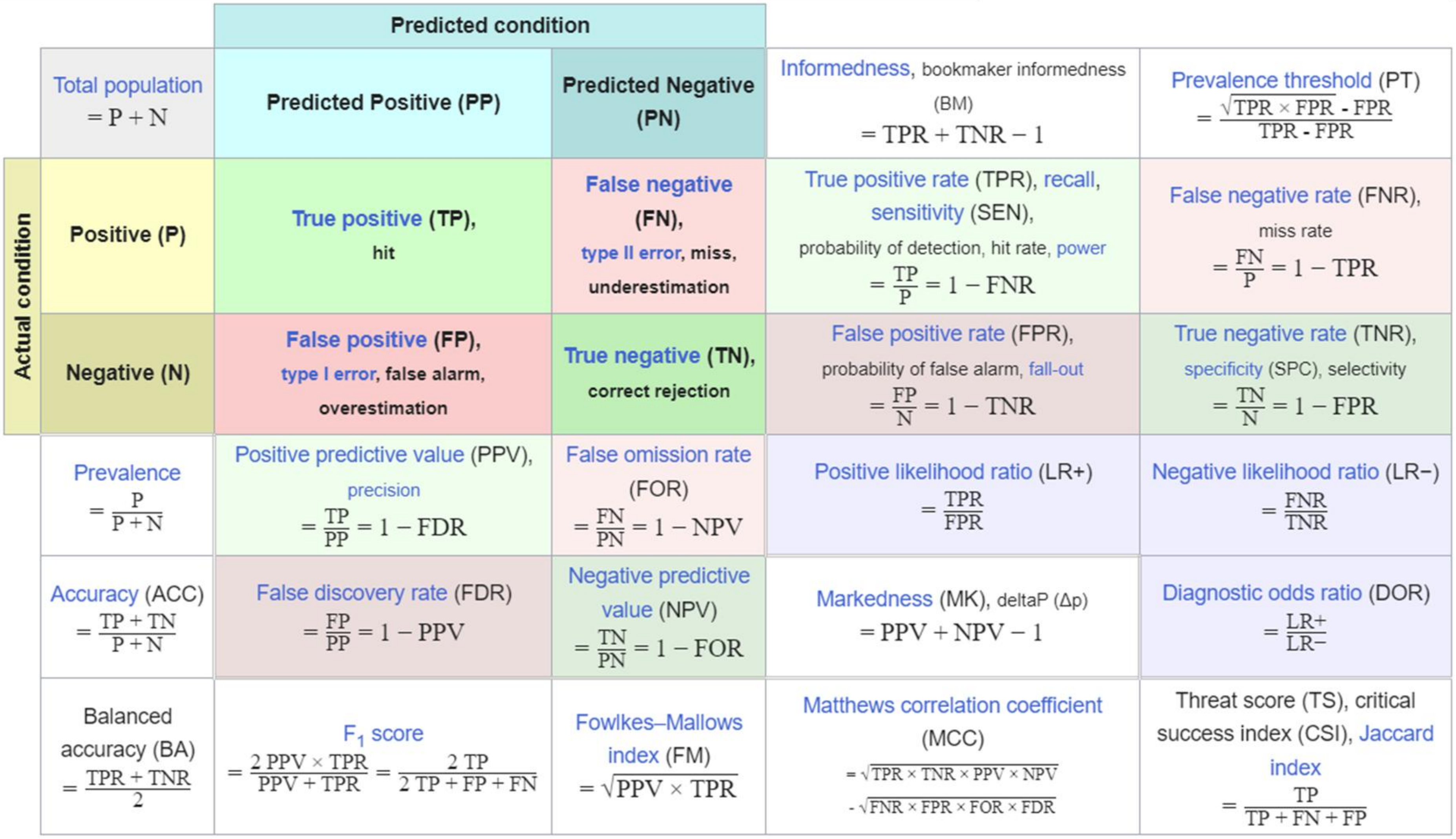
Enrichment in true actives at the top 1%-ranked molecules (EF1%)

$$\mathrm{EF1\%} = \frac{n_a/N_{1\%}}{n/N}$$

- n: total number of actives
- N: total number of ligands
- n_a: number of actives in the top 1%-ranked population
- $N_{1\%}$: 1% of the total population

BACK-UP SLIDE

Other metrics

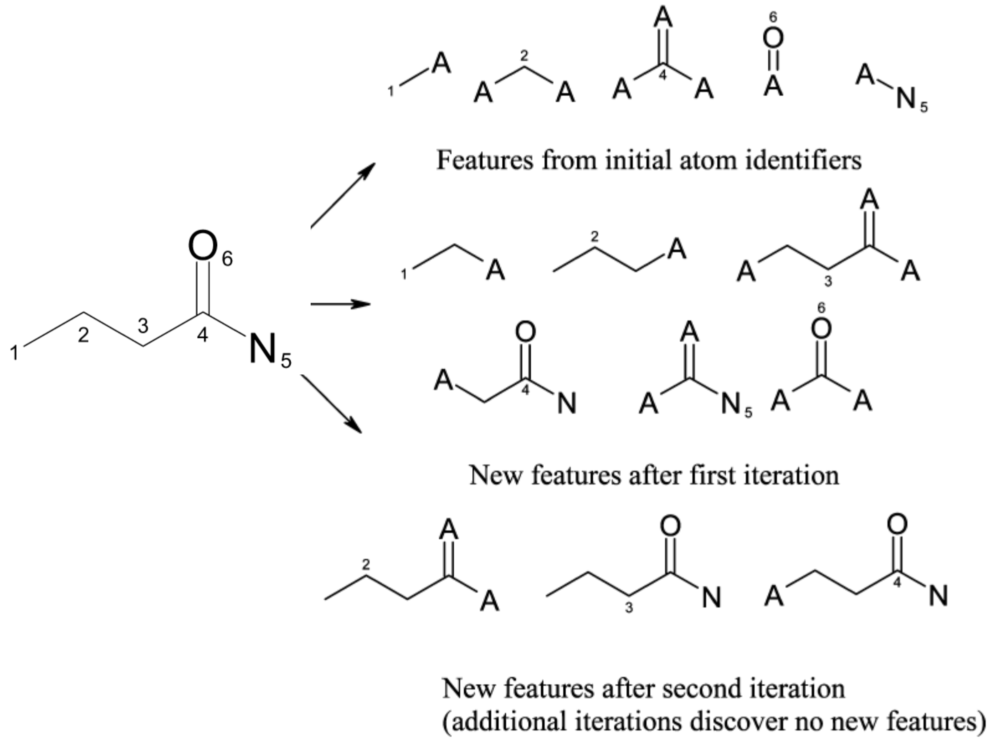


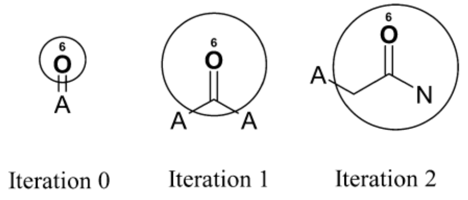
		Predicted condition			
	Total population $= \mathrm{P} + \mathrm{N}$	**Predicted Positive (PP)**	**Predicted Negative (PN)**	Informedness, bookmaker informedness (BM) $= \mathrm{TPR} + \mathrm{TNR} - 1$	Prevalence threshold (PT) $= \frac{\sqrt{\mathrm{TPR} \times \mathrm{FPR}} - \mathrm{FPR}}{\mathrm{TPR} - \mathrm{FPR}}$
Actual condition	**Positive (P)**	**True positive (TP), hit**	**False negative (FN), type II error, miss, underestimation**	True positive rate (TPR), recall, sensitivity (SEN), probability of detection, hit rate, power $= \frac{\mathrm{TP}}{\mathrm{P}} = 1 - \mathrm{FNR}$	False negative rate (FNR), miss rate $= \frac{\mathrm{FN}}{\mathrm{P}} = 1 - \mathrm{TPR}$
	Negative (N)	**False positive (FP), type I error, false alarm, overestimation**	**True negative (TN), correct rejection**	False positive rate (FPR), probability of false alarm, fall-out $= \frac{\mathrm{FP}}{\mathrm{N}} = 1 - \mathrm{TNR}$	True negative rate (TNR), specificity (SPC), selectivity $= \frac{\mathrm{TN}}{\mathrm{N}} = 1 - \mathrm{FPR}$
	Prevalence $= \frac{\mathrm{P}}{\mathrm{P} + \mathrm{N}}$	Positive predictive value (PPV), precision $= \frac{\mathrm{TP}}{\mathrm{PP}} = 1 - \mathrm{FDR}$	False omission rate (FOR) $= \frac{\mathrm{FN}}{\mathrm{PN}} = 1 - \mathrm{NPV}$	Positive likelihood ratio (LR+) $= \frac{\mathrm{TPR}}{\mathrm{FPR}}$	Negative likelihood ratio (LR−) $= \frac{\mathrm{FNR}}{\mathrm{TNR}}$
	Accuracy (ACC) $= \frac{\mathrm{TP} + \mathrm{TN}}{\mathrm{P} + \mathrm{N}}$	False discovery rate (FDR) $= \frac{\mathrm{FP}}{\mathrm{PP}} = 1 - \mathrm{PPV}$	Negative predictive value (NPV) $= \frac{\mathrm{TN}}{\mathrm{PN}} = 1 - \mathrm{FOR}$	Markedness (MK), deltaP (Δp) $= \mathrm{PPV} + \mathrm{NPV} - 1$	Diagnostic odds ratio (DOR) $= \frac{\mathrm{LR+}}{\mathrm{LR-}}$
	Balanced accuracy (BA) $= \frac{\mathrm{TPR} + \mathrm{TNR}}{2}$	F_1 score $= \frac{2\,\mathrm{PPV} \times \mathrm{TPR}}{\mathrm{PPV} + \mathrm{TPR}} = \frac{2\,\mathrm{TP}}{2\,\mathrm{TP} + \mathrm{FP} + \mathrm{FN}}$	Fowlkes–Mallows index (FM) $= \sqrt{\mathrm{PPV} \times \mathrm{TPR}}$	Matthews correlation coefficient (MCC) $= \sqrt{\mathrm{TPR} \times \mathrm{TNR} \times \mathrm{PPV} \times \mathrm{NPV}} - \sqrt{\mathrm{FNR} \times \mathrm{FPR} \times \mathrm{FOR} \times \mathrm{FDR}}$	Threat score (TS), critical success index (CSI), Jaccard index $= \frac{\mathrm{TP}}{\mathrm{TP} + \mathrm{FN} + \mathrm{FP}}$

BACK-UP SLIDE

Extended-connectivity fingerprints (ECFP)

2D descriptors in the form of bitstrings that represent all atoms of a given molecule by a set of atom identifiers that take into account all neighbors of each atom and the bonds that connect them.


Features from initial atom identifiers
New features after first iteration
New features after second iteration
(additional iterations discover no new features)



```
> <ECFP_0>     > <ECFP_2>     > <ECFP_4>
734603939      734603939      734603939
1559650422     1559650422     1559650422
-1100000244    -1100000244    -1100000244
1572579716     1572579716     1572579716
-1074141656    -1074141656    -1074141656
               863188371      863188371
               -1793471910    -1793471910
               -1789102870    -1789102870
               -1708545601    -1708545601
               -932108170     -932108170
               2099970318     2099970318
                              -87618679
                              1112638790
                              -627599602
```

D. Rogers and M. Hahn, *J. Chem. Inf. Model.* 2010, **50**, 742-54

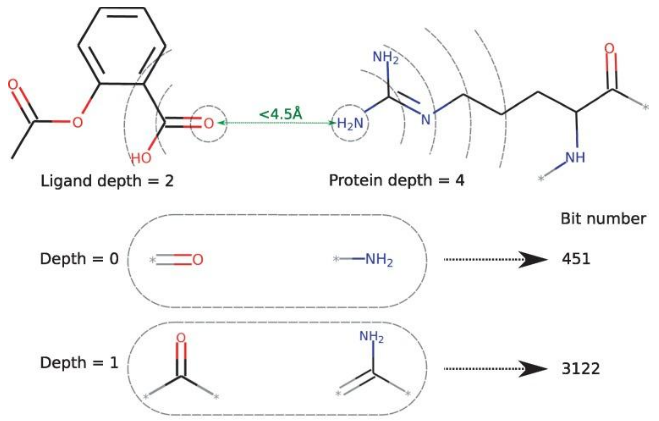
BACK-UP SLIDE

Protein-ligand extended-connectivity (PLEC) fingerprints

- Similar principle to that of ECFPs (only for interacting ligand and target atoms).
- Pairs of interacting target-ligand atoms are identified: pairwise distance ≤ ‘*distance_cutoff*’.
- ≥ 01 “environment(s)” are identified per atom in a pair, covering itself and its neighbors within an *n*-bond diameter (*n* = ‘*depth_protein*’, ‘*depth_ligand*’), registering its atomic features.
- Target and ligand environments of the same depth are paired up for an interacting target-ligand atom pair, all pairings are hashed and folded to a final fingerprint size (‘*size*’).
- We use the PLEC function from ODDT v0.7 to extract PLEC features.

```python
def parallel_plec_train(mol):
    feature = PLEC(mol, protein = receptor_train, size = 4092,
                   depth_protein = 4, depth_ligand = 2,
                   distance_cutoff = 4.5, sparse = False)
    return feature

def parallel_plec_test(mol):
    feature = PLEC(mol, protein = receptor_test, size = 4092,
                   depth_protein = 4, depth_ligand = 2,
                   distance_cutoff = 4.5, sparse = False)
    return feature
```



```python
num_cores = 20
train_features = Parallel(n_jobs = num_cores, backend = "multiprocessing")(delayed(parallel_plec_train)(mol) for mol in tqdm(train_mols))
test_features = Parallel(n_jobs = num_cores, backend = "multiprocessing")(delayed(parallel_plec_test)(mol) for mol in tqdm(test_mols))
```

BACK-UP SLIDE

Scoring functions

Empirical scoring functions:

- Using **physical/chemical terms** related to the binding of a ligand inside the binding pocket of a protein target
- **Coefficients** of contributing terms (H bonds, Hphobic contacts, rotatable bonds, vdW interactions) **pre-determined** based on existing complexes with known **experimental affinities** and 3D structures

Force field scoring functions:

- Estimating affinities by summing the strength of **non-covalent interactions** of a ligand (both intramolecular and intermolecular)
- Using parameters from classical force fields to predict the properties of molecules either in water or in a vacuum, **without using experimental affinity data**

Knowledge-based scoring functions:

- Based on **statistical observations** of **interacting atom pairs** in large 3D databases
- Computing the probability that the distance between a pair of atoms is equal to the value observed in a docking pose
- If an interacting atom pair occurs frequently → energetically favorable → favorable contribution to binding affinities

Machine-learning scoring functions

- **Not assuming a pre-determined functional form** for the relationship between binding affinities and structural features → functional form inferred directly from the input data (used for training)
- Computing many descriptors → which combination of parameters can reproduce experimental data?