Py-Pharm: An Open-Source Pharmacophore Based Virtual Screening Package Written in Python

SAPIENZA
UNIVERSITÀ DI ROMA

Facolta di Farmacia e Medicina, Ingegneria dell'informazione, informatica e statistica, Medicina e Odontoiatria, Scienze Matematiche, Fisiche e Naturali

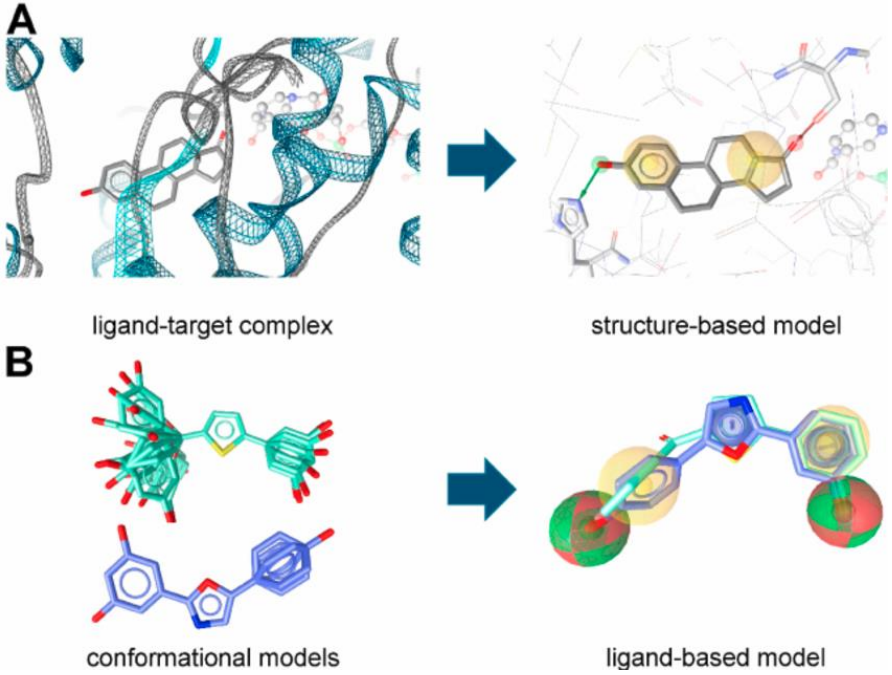
Corso di Laurea in Bioinformatica

Laureando: Julian Elijah Politsch
Matricula: 1831710

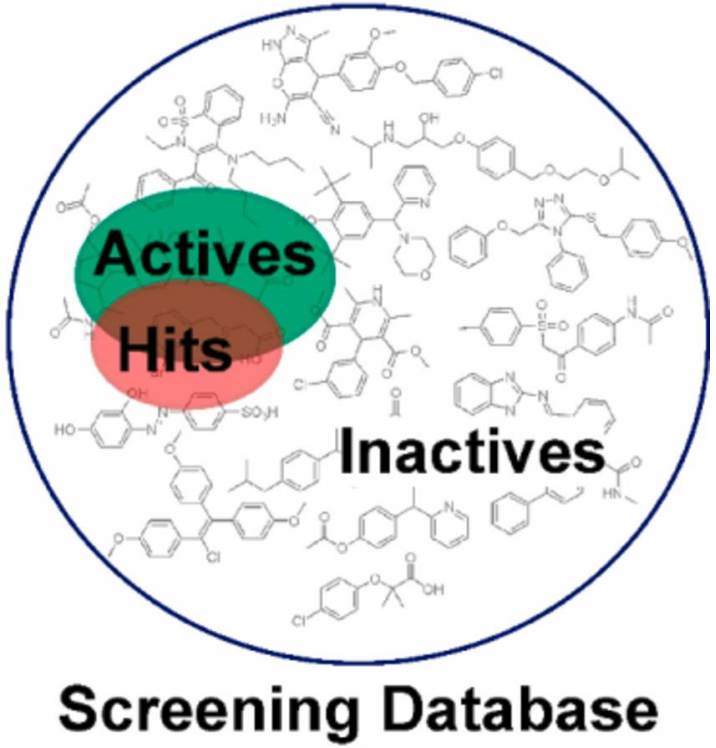
Relatore: prof. Rino Ragno

Pharmacophore

"A pharmacophore is the ensemble of steric and electric features that are necessary to **insure optimal supramolecular interactions** with a **specific biological target** and to trigger (or block) a biological response" -IUPAC



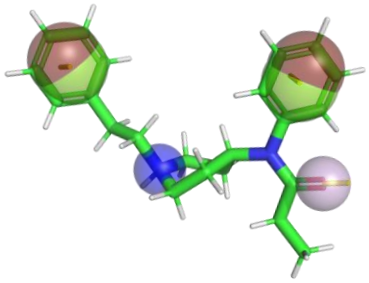
In CADD pharmacophore match indicates a potential hit compound



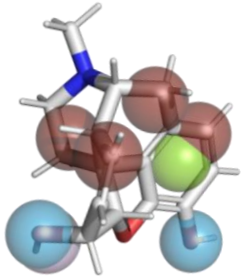
Multi-Ligand Pharmacophores

A **Shared Pharmacophore –** Minimum set of functional features shared by active ligands

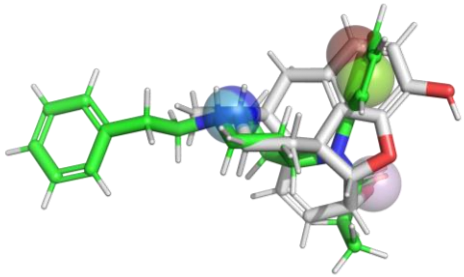
A **Merged Pharmacophore –** All functional features of active ligands



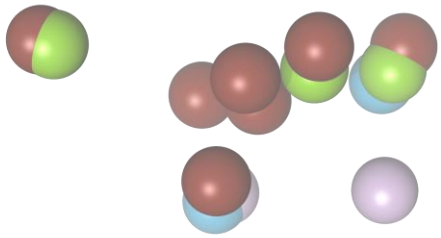
Fentanyl



Morphine



Shared Pharmacophore (PyPharm)



Merged Pharmacophore (PyPharm)

Scope of Thesis

Primary Goals:

1. **Design an open-source pharmacophore generation program for 3D-QSAR web server comparable to industry standard LigandScout**

2. **Create a virtual screening platform based on pharmacophore match theory with high accuracy**

Workflow Overview

Creating Shared Pharmacophore

Preprocessing-

Sub setting, Sanitizing, Conformational Analysis

↓

Pharmacophore Alignment

↓

Feature Ranking, Final Model Refinement

Virtual Screening

Hyperparameter Optimization

↓

Training Ensemble Machine Learning

↓

Hit Selection

Py-Pharm Package Contents

Completely customizable for knowledgeable user with example datasets

```
In [142]: ## Arguments, set them up for your system
          project = 'OpioidTest' #Name of project
          smiles = 'data/oppioidi(pochi).smi' #Comma seperated list of smile, numeric
          conf = 10 #Max number of conformations per molecule
          ref_select_method = 'Iterative' #Manual, Max_points, Iterative
          ref = '' #if ref_select_method is manual, Max_points is usually best unless
                   #also supports other common formats (SDF, Mol2)
          align_it = "/home/jepolitsch/.conda/envs/build/bin/align_it" #Path to exect
          shape_it = "/home/jepolitsch/.conda/envs/shape-it/bin/shape-it"
          scores = "Empty"
```

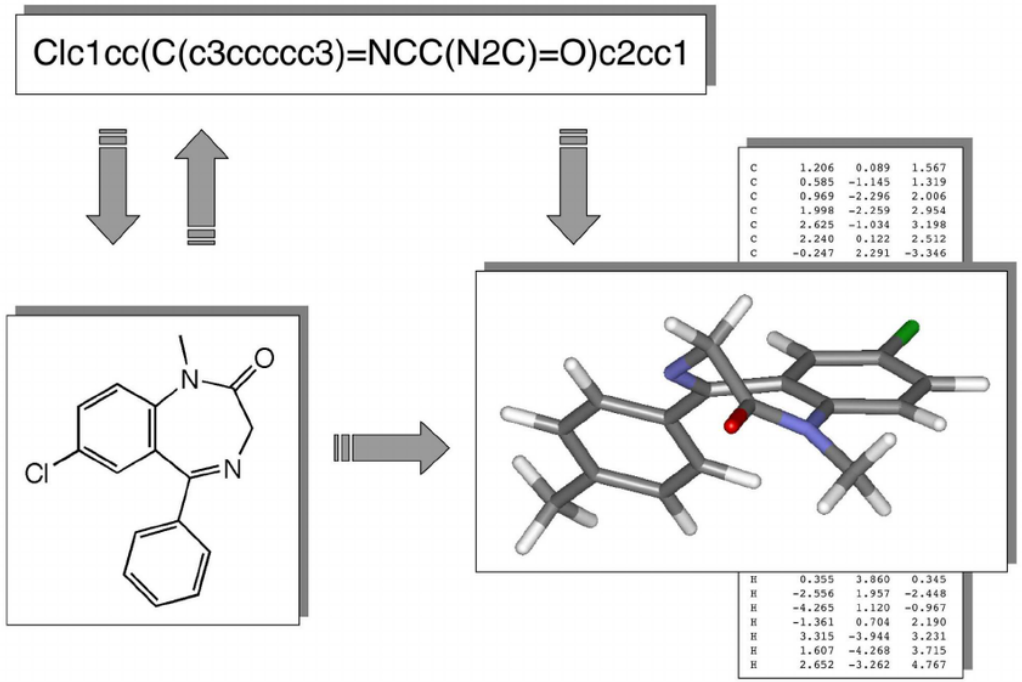
Optional Arguments:

```
In [143]: training3D = ''
          cutoff = '0'
          merged = False
          silent = True
          noHybrid = True
          epsilon = '0.5'
          pypharm.setup_variables(project, smiles, conf_G=conf, merged_G=merged,
                                  noHybrid_G= noHybrid, epsilon_G=epsilon)
          #Set global variables in PyPharm.py
```

```
PyPharm
├── Jupyter_PyPharm_1.0.6.ipynb
├── PyPharm.py
├── Virtual_Screening_1.0.1.ipynb
└── data
    ├── CFC_andrea_conH.smi
    └── antiepilettici.smi
```

Input for Py-Pharm:

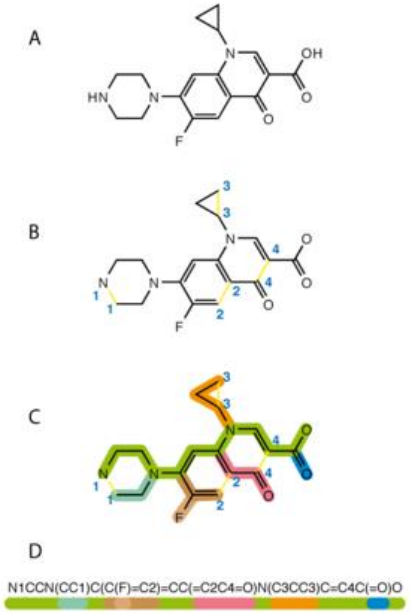
SMILES = **S**implified **M**olecular-**I**nput **L**ine-**E**ntry **S**ystem



26 Million Compounds:

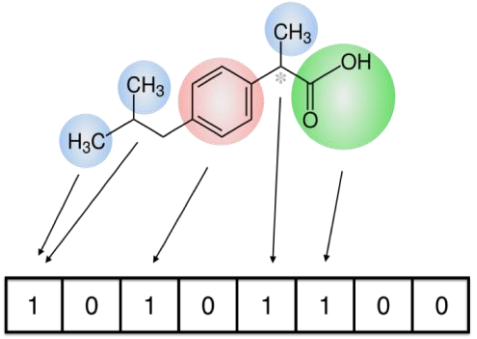
= 6 **GB** as SDF (3D)

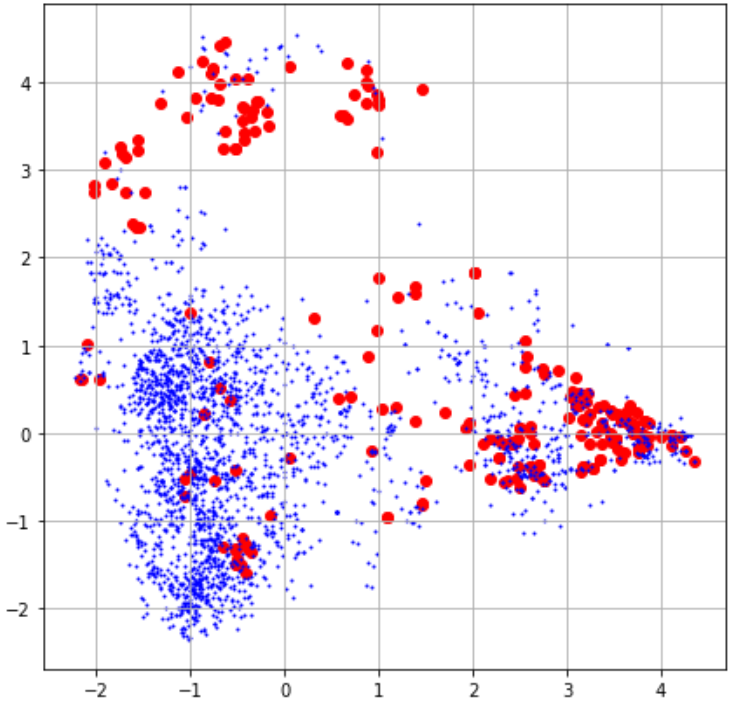
= 300 **MB** as SMILES

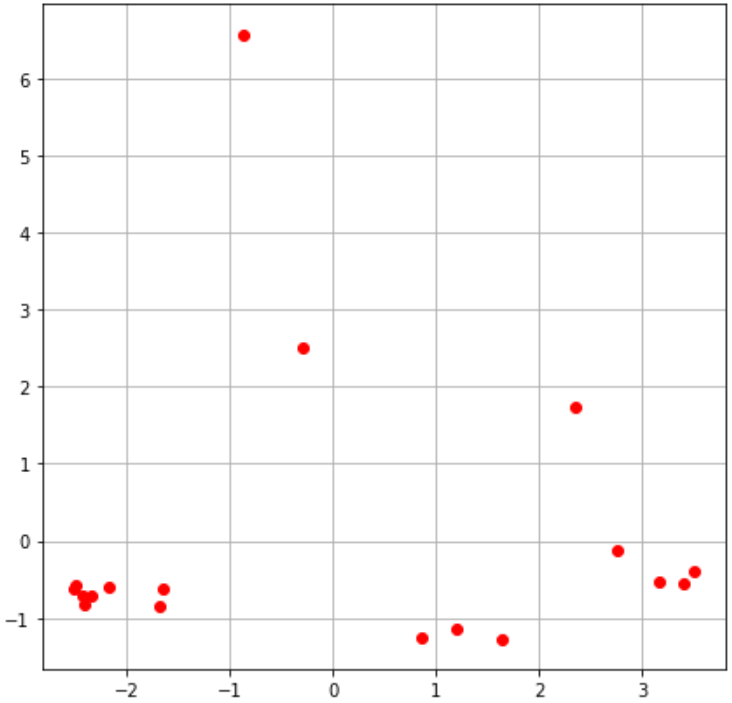


Preprocessing

1. Selection of highly active molecules





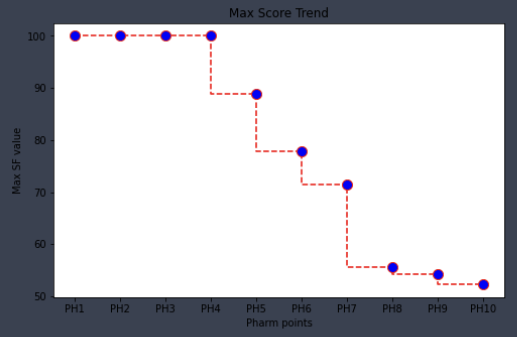


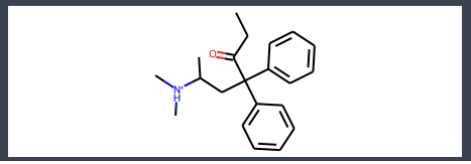
Preprocessing

1. Selection of highly active molecules
2. Sanitize SMILES (Duplicates, Salts, etc.)
3. Protonation at desired pH
4. Reference Selection (Structure to align all other probes to)

```
In [151]: ### - 1) Select Reference based on specified parameters

          smiles = pypharm.preprocess_smiles("data/oppioidi(pochi).smi") #Sanitize Sm

          ref, scores = pypharm.select_ref("Iterative", smiles, pharm_scores=scores)

          scoring_plot(scores)

          Chem.MolFromSmiles(open(ref,'r').read().split(",")[0])
```

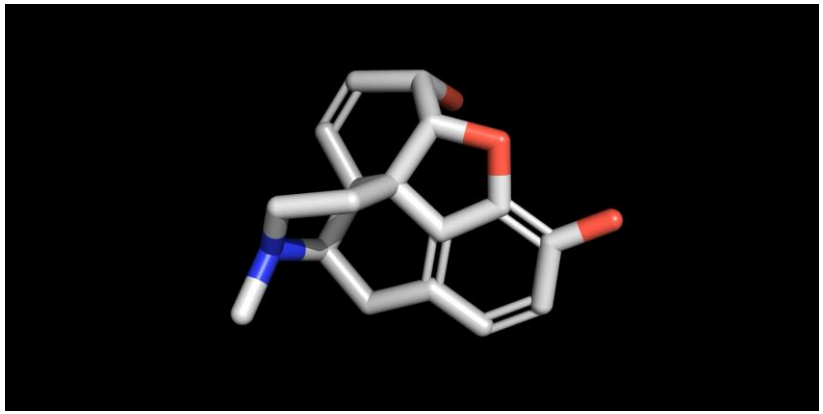
```
Sanitization of data/oppioidi(pochi).smi: 100% | 9/9 [00:00<00:00, 347.56it/s]

removed 0 invalid or duplicate smiles
Select Number of Points (1 to 10): 7
 methadone selected as ref. with score 71.4286 for 7 point pharmacophore
```





Preprocessing

1. Selection of highly active molecules
2. Sanitize SMILES (Duplicates, Salts, etc.)
3. Protonation at desired pH
4. Reference Selection
5. MMFF Conformational Analysis

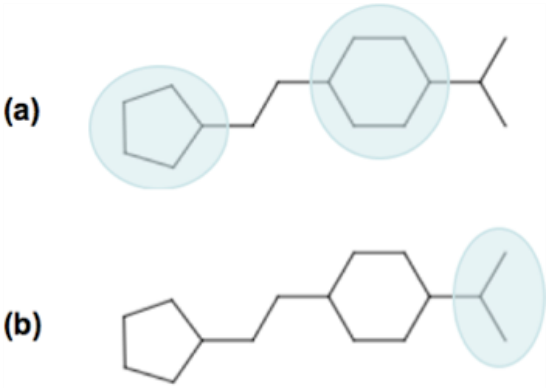


```
In [152]: # - 2) Perform conformational analysis for the reference SMILE and the training set

ref3D, refEnergy = pypharm.gen3D(ref,str(project_path+"_ref.sdf"), '1', max_iter=200)
training3D, trainingEnergy = pypharm.gen3D(smiles, str((project_path + "_gen3D.sdf")), conf)

100% 1/1 [00:00<00:00, 9.50it/s]

SDF file written for OpioidTest/OpioidTest_ref.smi  in OpioidTest/OpioidTest_ref.sdf

100% 9/9 [00:04<00:00, 1.56it/s]

SDF file written for data/oppioidi(pochi)_sanitized.smi  in OpioidTest/OpioidTest_gen3D.sdf
```

Alignment Phase

Feature Mapping

- Mapping of two pharmacophores consists of a list of points from **A** and **B** in which corresponding points have a **compatible functional group**

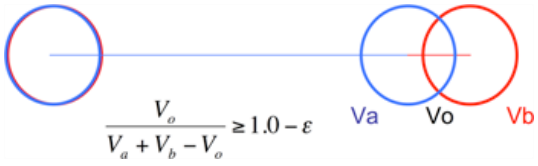


Alignment

- Grant/Pickup algorithm with pharmacophores
- Combination of gradient-ascent [$T(X,Y,Z)$] and rigid-body rotation [$R(\phi,\theta,\psi)$], the maximal volume overlap is determined.

Alignment Scoring

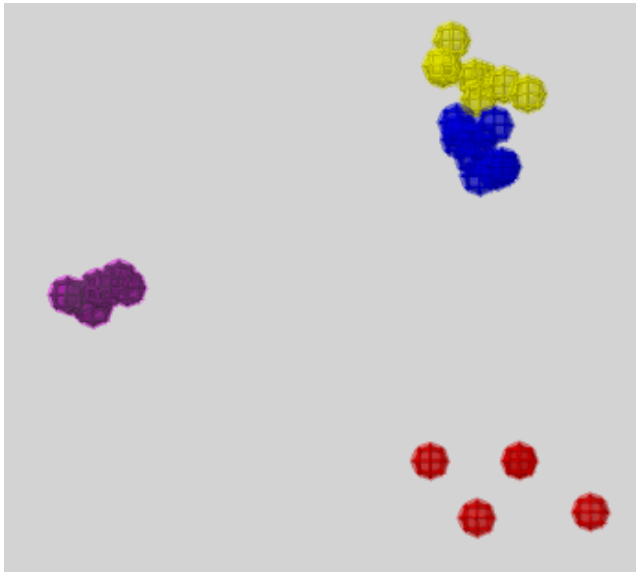
- **Tanimoto** = V_O / [V_A + V_B – V_O]



Feature Ranking

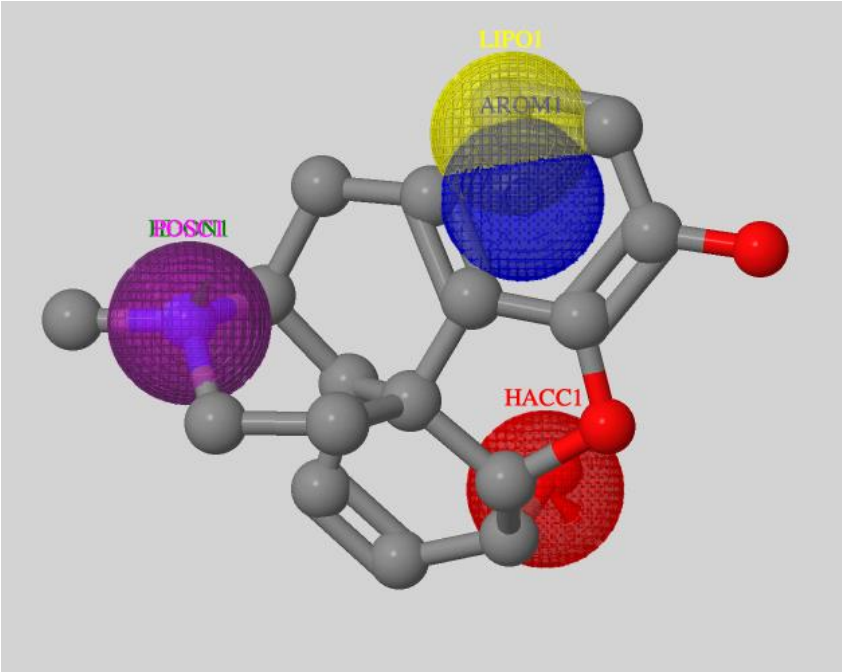
1. Features are ranked by highest conservation
2. User chooses desired number of features

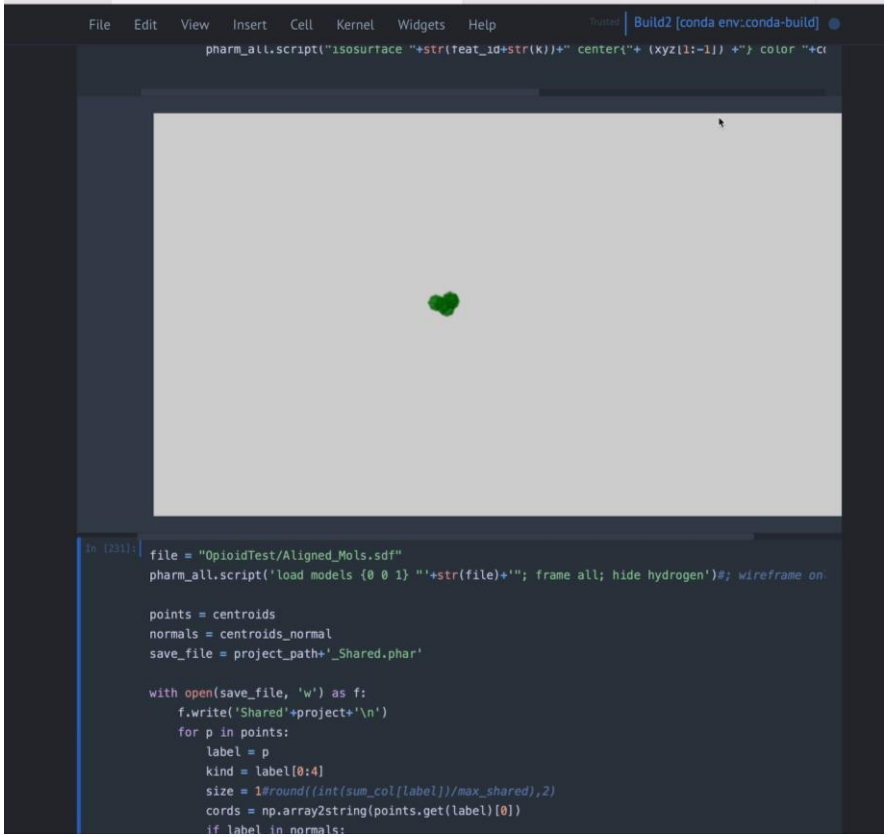
	AROM1	HDON1	HDON2	HDON3	HACC1	LIPO1	POSC1
morphine.0	1	1	1	1	1	1	1
meperidine.5	1	1	0	0	1	1	1
methadone.26	1	1	0	0	0	1	1
oxycodone.8	1	1	0	0	1	1	1
fentanyl.20	1	1	0	0	0	1	1
pentazocine.28	1	1	1	0	0	1	1
tramadol.27	1	1	0	0	0	1	1
ketobemidone.24	1	1	0	0	0	1	1
diphenoxylate.12	1	1	0	0	1	1	1
Sum	9	9	2	1	4	9	9



Final Model Creation

1. Features are ranked by highest conservation
2. User chooses desired number of features
3. Centroids represent shared features





Workflow Overview

Creating Shared Pharmacophore

Preprocessing-
Sub setting, Sanitizing, Conformational Analysis

Pharmacophore Alignment

Feature Ranking, Final Model Refinement

Shared Pharm

Virtual Screening

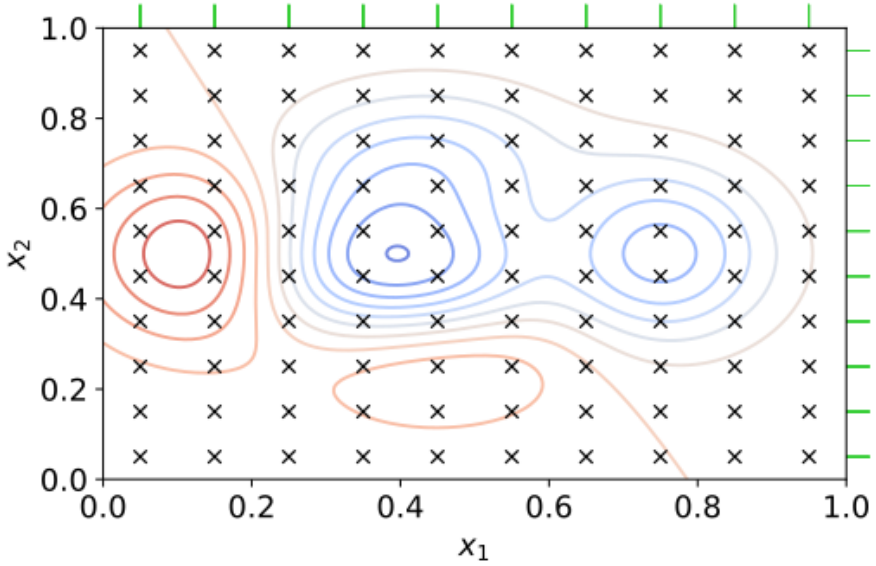
Hyperparameter Optimization

Training Ensemble Machine Learning

Hit Selection

Hyperparameters

1. Molecules of known activity are subset into Train (70%) and Test (30%)
2. Hyperparameters optimized based by grid search

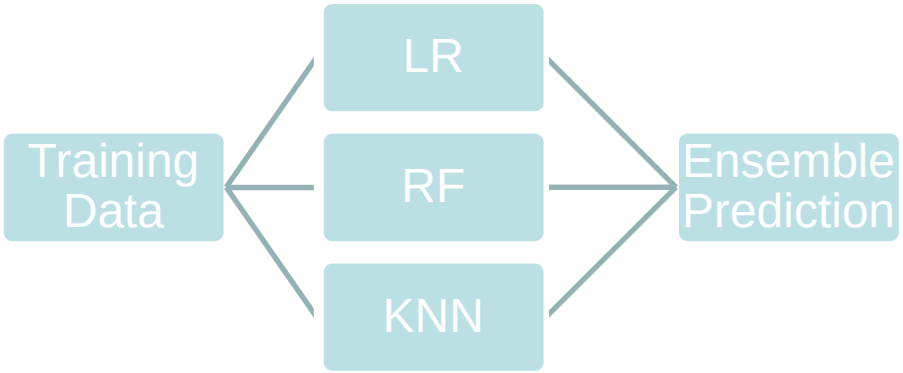



```
parameters_lr = {
 'C': np.arange(0.1, 1.5, 0.2),
 'fit_intercept': [True, False],
 'solver': ['newton-cg', 'saga'],
 "multi_class" : ['auto', 'ovr', 'multinomial']
}
parameters_rf = {
    "n_estimators": np.arange(10, 500, 50),
    "criterion" : ["gini", "entropy"],
}

parameters_knn = {
 'algorithm': ['auto', 'ball_tree', 'kd_tree', 'brute'],
 'n_neighbors': np.arange(1,80, 4),
 'weights': ['uniform', 'distance']
}

gslr = model_selection.GridSearchCV(estimator=clf1, param_grid=parameters_lr,
                                    scoring='accuracy', cv=cv, n_jobs=8)
gsrf = model_selection.GridSearchCV(estimator=clf2, param_grid=parameters_rf,
                                    scoring='accuracy', cv=cv, n_jobs=8)
gsknn = model_selection.GridSearchCV(estimator=clf3, param_grid=parameters_knn,
                                    scoring='accuracy', cv=cv, n_jobs=8)
```

Training Machine Learning

1. Molecules of known activity are subset into Train (70%) and Test (30%)
2. Hyperparameters optimized based by grid search
3. Combinational soft voting to choose best methods between
   - Logistic Regression
   - Random Forest
   - K Nearest Neighbors

```
In [34]: #Soft Voting Prediction
clf1 = KNeighborsClassifier(n_jobs=8, n_neighbors=1)
clf2 = rf_params
clf3 = knn_params

X = training_mat.drop("Active", axis=1)
y = training_mat["Active"]

scaler = StandardScaler()
scaler = scaler.fit(X)
X = scaler.transform(X)

score_validation_scaled = scaler.transform(score_validation)
score_test_set_scaled = scaler.transform(score_test_set)

eclf1 = VotingClassifier(estimators=[
    ('rf', clf2), ('knn', clf3)], voting='hard', n_jobs=8)

eclf1 = eclf1.fit(X, y)

eclf2 = VotingClassifier(estimators=[
    ('rf', clf2), ('knn', clf3)], voting='soft', n_jobs=8)

eclf2 = eclf2.fit(X, y)

clf1 = clf1.fit(X, y)
clf2 = clf2.fit(X, y)
clf3 = clf3.fit(X, y)
```



Assessment of Methodology

Case Study of Erα: Retrospective Analysis

- 17 most active compounds ($pIC_{50} > 9$) for Shared
- 20685 inactive decoys (Negative Test)
- 382 active compounds (Positive Test)
- 100 conformations generated

D U D • E

A Database of Useful Decoys: Enhanced

Decoys: similar physico-chemical properties but dissimilar 2-D topology.

Indones. J. Chem., 2021, 21 (1), 137 - 147

Ligand Based Pharmacophore Modeling, Virtual Screening, and Molecular Docking Studies of Asymmetrical Hexahydro-2H-Indazole Analogs of Curcumin (AIACs) to Discover Novel Estrogen Receptors Alpha (ERα) Inhibitor

Hariyanti[1], Kusmardi[2], Arry Yanuar[3], and Hayun[3,*]

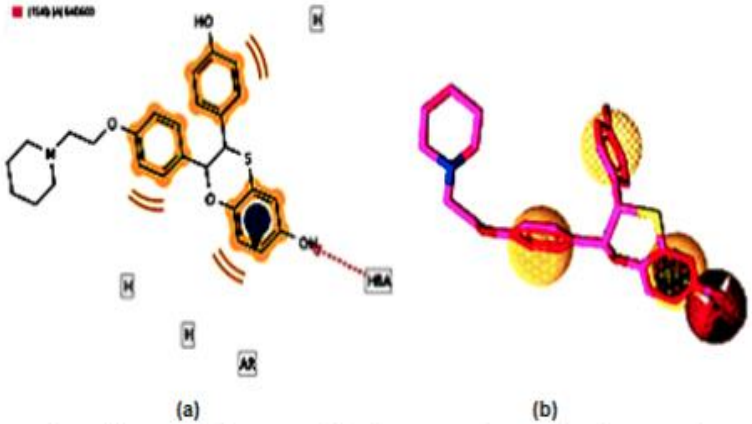
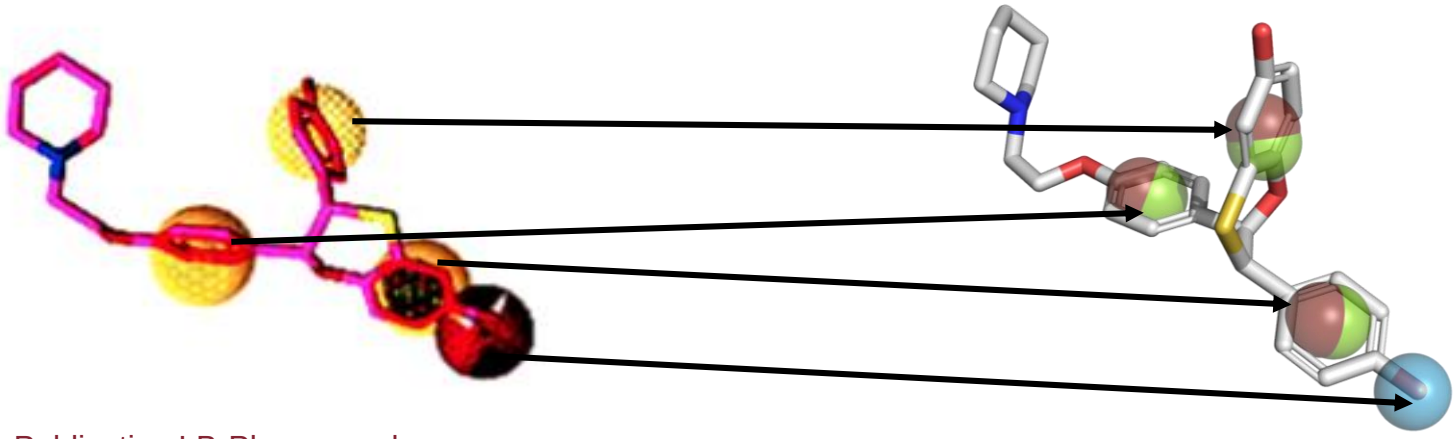


Fig 2. Pharmacophore (a) 2D and (b) 3D models of E4D600 obtained by the LigandScout 4.2 software

Case Study of ERa

Py-Pharm use's **Hybrid Points** because Aromatic and Hydrophobic regions occure together



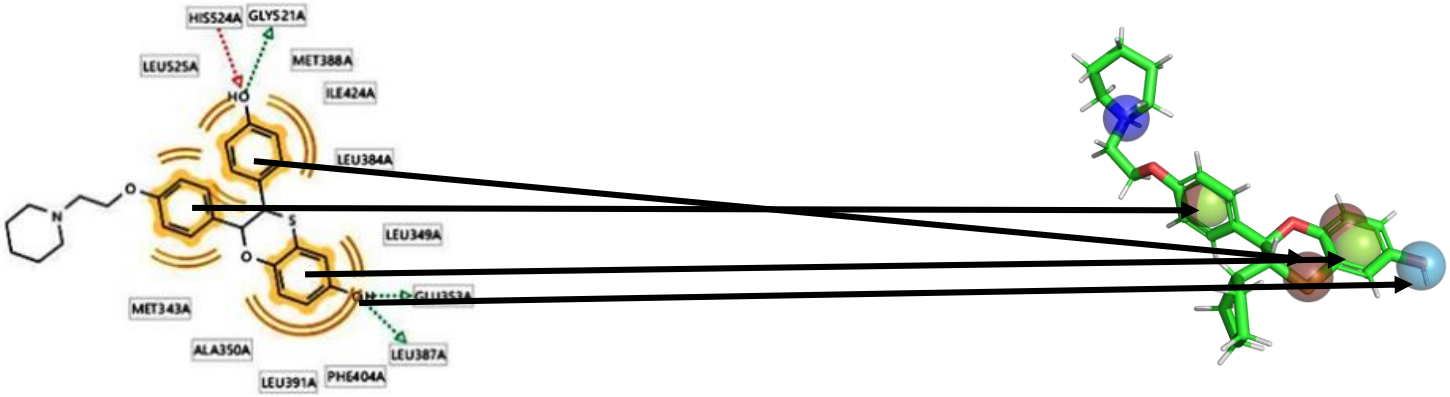
Publication LB-Pharmacophore

Py-Pharm Shared Pharmacophore (PH4)

3	(YELLOW) HYDROPHOBIC INTERACTION (RED)	3
1	AROMATIC INTERACTION (GREEN)	3
0	POSITIVE CHARGE CENTER	0
1	(RED) HYDROGEN BOND ACCEPTOR	0
0	HYDROGEN BOND DONOR(BLUE)	1

Case Study of ERa

Interestingly, the structure-based pharmacophore had a H-Bond Donor instead of acceptor, aligning with Py-Pharms model



Publication SB-Pharmacophore

Py-Pharm Shared Pharmacophore 6 points

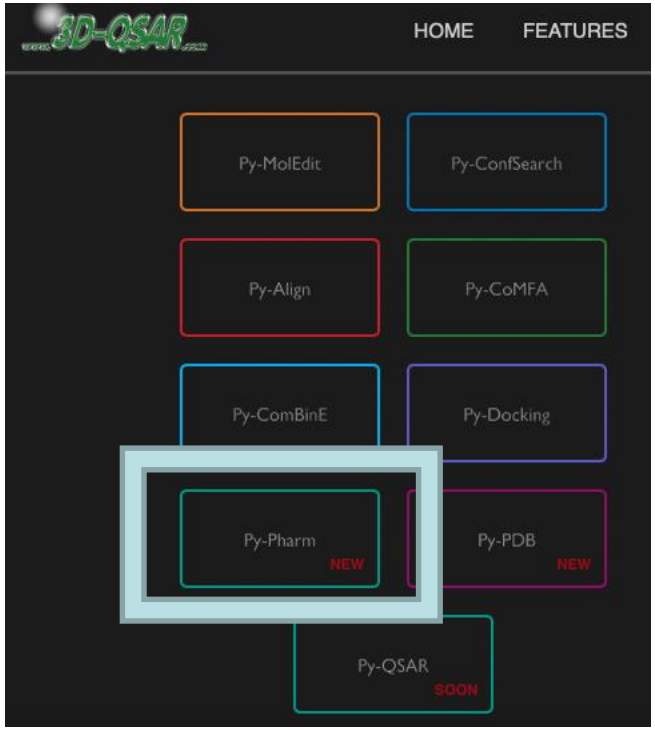
3	(YELLOW) HYDROPHOBIC INTERACTION (RED)	3
1	AROMATIC INTERACTION (GREEN)	2
0	POSITIVE CHARGE CENTER (DARK BLUE)	1
1	(RED) HYDROGEN BOND ACCEPTOR	0
2	(GREEN) HYDROGEN BOND DONOR(BLUE)	1

Primary Goals:

1. **Design an open-source pharmacophore generation program for 3D-QSAR web server comparable to industry standard LigandScout**

2. **Create a virtual screening platform based on pharmacophore match theory with high accuracy**

Results



Case Study of ERa

Py-Pharm was used to screen the same set of decoys and active molecules as published:

Observations: Lower sensitivity, Higher specificity and Accuracy

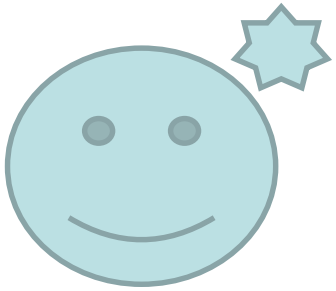
Measure	Publication	Py-Pharm	Derivations
Sensitivity	0.687	0.49	TPR = TP / (TP + FN)
Specificity	0.845	0.95	SPC = TN / (FP + TN)
Accuracy	0.84	0.94	ACC = (TP + TN) / (P + N)
Fl Score	0.13	0.21	Fl = 2TP / (2TP + FP + FN)
Matthews Correlation Coefficient	0.18	0.23	TP*TN - FP*FN / sqrt((TP+FP)*(TP+FN)*(TN+FP)*(TN+FN))

Primary Goals:

1. Design an open-source pharmacophore generation program for 3D-QSAR web server comparable to industry standard LigandScout

2. **Create a virtual screening platform based on pharmacophore match theory with high accuracy**

Results



Thank you!

Julian Politsch

Email: politsch.1831710@studenti.uniroma1.it

References

[1] Grant, J.A.; Gallardo, M.A.; Pickup, B.T. (1996) 'A fast method of molecular shape comparison: a simple application of a Gaussian description of molecular shape', *J. Comp. Chem.* **17**, 1653-1666 [wiley/19961115]

[2] Taminau, J.; Thijs, G.; De Winter, H. (2008) 'Pharao: pharmacophore alignment and optimization', *J. Mol. Graph. Model.* **27**, 161-169 [pubmed/18485770]

[3] Indonesian Journal of Chemistry (ISSN 1411-9420 / 2460-1578) - Chemistry Department, Universitas Gadjah Mada, Indonesia.