Tutorial of Fingerprint-based Compound Classifier

Introduction to the Theory and Implementation of LiCABEDS

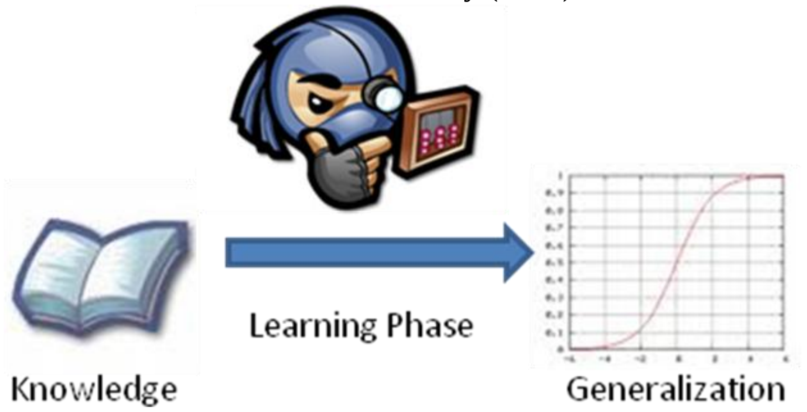
Chao Ma

Aug-05-2010

Introduction

Ligand property prediction has become an essential component in modern computer-aided drug design. Many "lead" candidate compounds are denied at later phases of clinic trials due to the lack of certain physiochemical or pharmacological properties. Thus, besides virtual screening, it is significantly beneficial to consider ligand properties in the early stage of a drug discovery project. Quantitative structure-property relationship (QSPR) is a popular approach to predict ligand property and refine compound structure in order to get better pharmacological profile. In QSPR, ligand property is correlated to certain key fragments or structural patterns through a quantitative procedure. Up to now, QSPR has been successfully applied to model compound solubility, CLogP, ADME/T, mutagenicity, carcinogenicity, and so on. Although some heuristic rules are well accepted in drug discovery industry, such as Lipinski's Rule of Five for drug likeness, most quantitative structure-property relationship is created through statistical inference or pattern recognition techniques. The recent advance in the world of artificial intelligence offers an excellent opportunity for chemists and pharmacologists to rationally design and develop candidate compounds regarding biological activity and pharmacological property.

Traditionally, ligand property prediction can be made by a predictive model that is trained on a knowledge base using supervised learning methodology. In supervised learning, a set of labeled compounds are presented to a learner (e.g. a computer). These labeled compounds are usually annotated with meaningful biological or biochemical value, such as inhibitor constant and acute toxicity (LD_{50}).



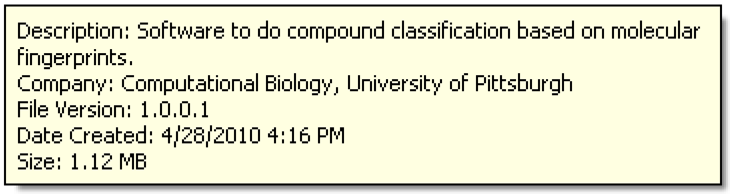
As illustrated by the figure above, a predictive model is "learned" from some labeled training samples (e.g. tested compounds) and a predictive model is generalized through the pattern observed in the training dataset.

In the program, fingerprint-based compound classifier (FCC), established models can be imported into a project to make predictions. However, the program still allows users developing their own models from customized training compound datasets in order to tackle various ligand properties. In the following sections, a sample walk-through project (Appendix) is provided to guide readers to understand the workflow of FCC. The basic principle of the embedded classification algorithm, LiCABEDS, is briefly introduced as well.

FCC is developed as “green” software for Microsoft Windows system, which means that installation is not strictly required. To launch the program, locate the executable file and double click the “FCC.exe” icon.

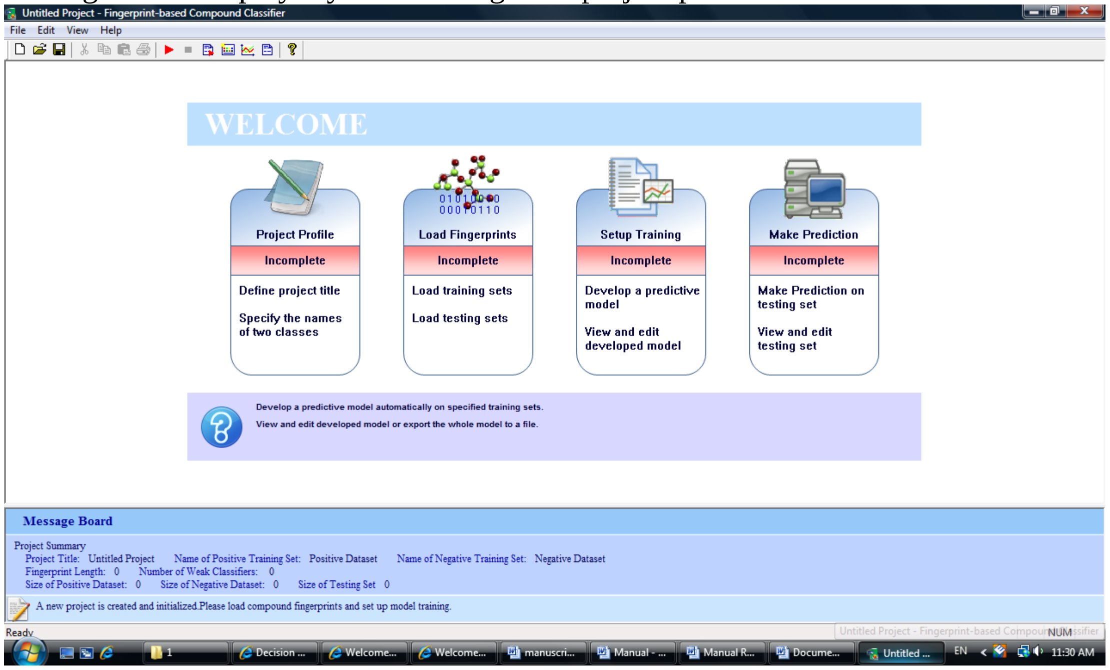
FCC.exe
Software to do compound clas...
Computational Biology, Univer...

FCC.pdb
PDB Files
6,155 KB



Description: Software to do compound classification based on molecular fingerprints.
Company: Computational Biology, University of Pittsburgh
File Version: 1.0.0.1
Date Created: 4/28/2010 4:16 PM
Size: 1.12 MB

The interface of the program is displayed in the figure below. Similar to most Window programs, FCC also has a title bar, showing the title of current project and “minimize, maximize, close” buttons. It is followed by a menu bar and a tool bar. The tool bar provides shortcuts to some frequently used functions, such as setting up a new project, copy-and-paste functions and model training. Besides toolbar, the whole list of program functions can be found in the menu. A main window is designed for user-computer information exchanging. In the screenshot, the main window displays welcome page, project status and short introduction to each module. Underneath the welcome page, message board displays system message and project parameters.



Define project title

Specify the names of two classes

Load training sets

Load testing sets

Develop a predictive model

View and edit developed model

Make Prediction on testing set

View and edit testing set

Develop a predictive model automatically on specified training sets.
View and edit developed model or export the whole model to a file.

Message Board

Project Summary
Project Title: Untitled Project Name of Positive Training Set: Positive Dataset Name of Negative Training Set: Negative Dataset
Fingerprint Length: 0 Number of Weak Classifiers: 0
Size of Positive Dataset: 0 Size of Negative Dataset: 0 Size of Testing Set 0

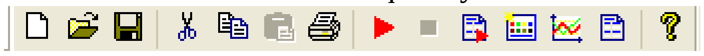
A new project is created and initialized.Please load compound fingerprints and set up model training.

In FCC, a ligand classification project mainly consists of four steps:

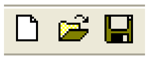
1. Define a project in “Project Profile”: name the title of the project and specify ligand categories of interest.
2. Load Fingerprints: import training and testing compound datasets in predefined fingerprint format.
3. Setup Training: develop a LiCABEDS model
4. Make Predictions: use the developed model to make predictions on testing compound dataset.

The homepage also tells the progress of current project by showing different colors on each tag. A module tag will turn green after a step is finished, e.g. model training, corresponding. A red tag indicates this step is incomplete whereas a yellow tag indicates this step is partially finished.

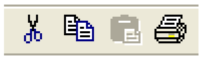
The FCC toolbar includes frequently accessed functions:



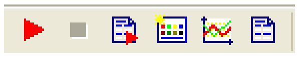
The function of each button is listed below. Detailed introduction will be given in following chapters.

File functions: 

1. New project: current project profile and data are restored to the original settings. All the changes made to the project are discarded. A warning message will display to remind saving your work.
2. Open project: a previous saved workspace will be restored. All the changes made to the current project are discarded.
3. Save project: all the project settings and datasets will be saved to a FCC file on the hard drive. The project will resume next time by opening the FCC workspace file.

Editing Functions: 

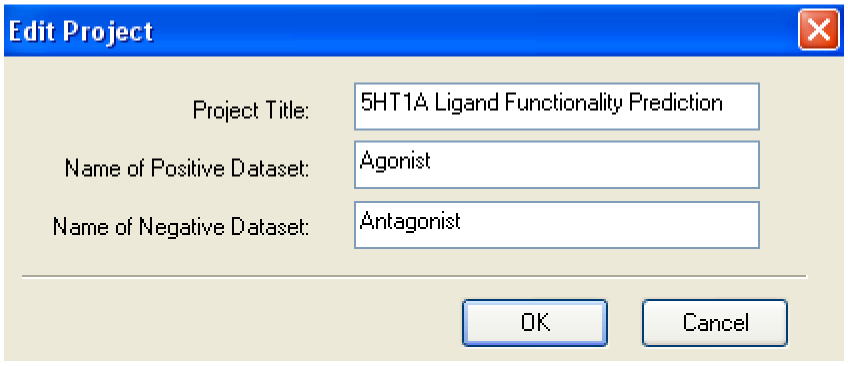
1. Cut: delete the selected items and deposit the content into system clipboard. The button turns grey if the action is not applicable.
2. Copy: Deposit the selected items into system clipboard. The button turns grey if the action is not applicable.
3. If a compatible format is available in the clipboard, the copied contents will be pasted to the current project. This function only works in prediction window.

Project Functions: 

1. Start a new model training or continue with previous model training
2. Stop undergoing model training.
3. Make predictions using developed predictive model
4. Go to home page
5. Go to training window for model browsing
6. Go to prediction window to view results.

<u>Step One: Define a Project</u>

A new project starts with defining its aim and two compound categories. Click the "Project Profile" or go to "Edit Project Title" in "Edit" menu to define a project. A dialog box will pop up and its interface is displayed below:

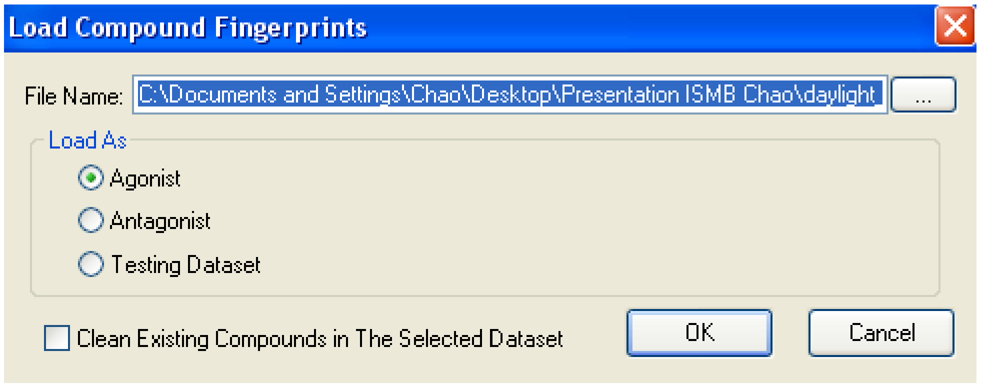


Project Title: 5HT1A Ligand Functionality Prediction
Name of Positive Dataset: Agonist
Name of Negative Dataset: Antagonist

In "Project Title", input a name to distinguish the current project from others. The title name will become the default file name in workspace saving (to be discussed later). The project title is also displayed in the "Message Board" in the program, so that user can easily identify the aim and content in this project. Next, specify two categories. For convenience, the categories could be the properties to be modeled, e.g. agonist and antagonist in the screenshot. The program will automatically output the predicted category for each testing compound after predictions are made. Click "OK" button to finish project definition. Users can always repeat the same procedure to change the project settings. User may start a new project by clicking 🗋 button on the toolbar or go to menu "File->New Project…".

Step Two: Load Fingerprints

The imbedded classification algorithm in FCC is called Ligand Classifier of Adaptively Boosting Ensemble Decision Stumps (LiCABEDS). The theory and performance of LiCABEDS are discussed in the published manuscript. Like many other supervised learning algorithms, LiCABEDS also relies on training compounds to derive a predictive model that will be used to make prediction on testing datasets.



File Name: C:\Documents and Settings\Chao\Desktop\Presentation ISMB Chao\daylight

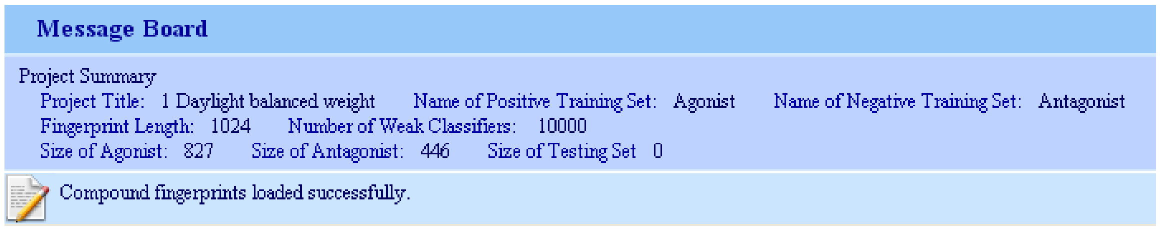
Load As
[✓ Agonist]
Antagonist
Testing Dataset

☐ Clean Existing Compounds in The Selected Dataset

The screenshot above shows the interface for compound dataset importing. In FCC, compounds are represented molecular fingerprints instead of structure information. Molecular fingerprints define the hypothesis space, within which possible decision stumps in LiCABEDS can be generated (refer to publication for details). A sample fingerprint is given below:

```
L011394 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 1 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0
L001401 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 1 1 0 0 0 0 0 1 0 0 0
L005495 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 0 0 0 0 1 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
L008641 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
L008820 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 1 0 0 0 1 0 0 0 0 0 0 0 0 0 0 1 0 1 0 0 0 0 0
L012936 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0
L004095 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 1 1 0 0 0 0 0 0 0 0 0 1 0 1 0 0 1 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 1 0 1 1 0 0 0 0 0
L004092 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 1 0 1 0 0 1 0 0 0 1 0 0 1 0 0 0 0 0 0 0 0 1 0 1 1 0 0 0 0 0
```

In this format, each line defines a compound. The first field of every row is the index of training or testing compounds. It is then followed by fingerprints in 0/1 binary format. The compound index and fingerprint values are separated by single space. The protocol to generate Molprint2D fingerprint is given in the appendix. To import compound datasets, select the category, which the dataset will be imported to (e.g. agonist in the picture). Then click "…" button to locate the fingerprint file. Compound datasets can be additively imported to the same category several times. However, users have the option to check "Clean Existing Compounds in The Selected Dataset". Once this option is checked, the data in the selected category will be overwritten by the specified fingerprint file. Both categories of training compounds have to be loaded before model training starts. Testing compound dataset could also be loaded at the same time, so that predictions can be made after model development. Note that the data needs to comply with the fingerprint specification mentioned above. Sample training and testing datasets are available for downloading. Refer to the simple walk-through project in Appendix. Message Board will display the updated status of the project after data is successfully imported:


Message Board

Project Summary
Project Title: 1 Daylight balanced weight   Name of Positive Training Set: Agonist   Name of Negative Training Set: Antagonist
Fingerprint Length: 1024   Number of Weak Classifiers: 10000
Size of Agonist: 827   Size of Antagonist: 446   Size of Testing Set 0

Compound fingerprints loaded successfully.

A brief introduction to molecular fingerprint and its application for data mining

What is molecular fingerprint

Molecular fingerprint was originally developed for similarity search and high speed structural screening. It encodes molecular structure in a series of binary digits that represent the presence and absence of particular substructures in the molecule. Just as particular genes in the human body determine people's physical appearance, these molecular fingerprints inform the computer of where to categorize the unknown subjects based on their fingerprints or appearance. For instance, consider an incoming email as an analogy to molecular fingerprints. Before the email arrives, several categories can be listed in a row such as blue, red, tall, short, soft, and rough. The computer can then "fingerprint" the incoming email. For instance if the incoming mail was blue, tall, and soft its molecular fingerprint would appear to be:

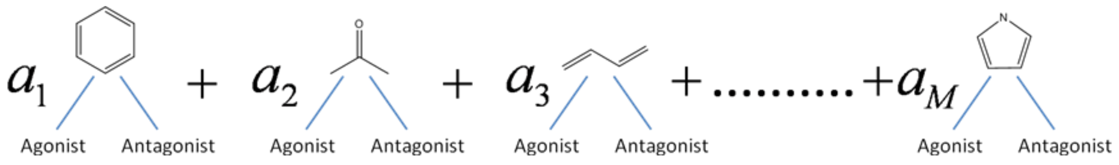
Blue	Red	Tall	Short	Soft	Rough
1	0	1	0	1	0

Therefore, 1 indicates that the subject does contain that particular characteristic where as a 0 indicates that the subject does not contain that particular characteristic. Ideally, the computer would receive more than one such "mail" and would then classify them into their appropriate files according to their fingerprints. More specifically, the molecular fingerprints in this type of machine learning will be to categorize the compound based on its particular chemical structure. Therefore the molecular fingerprints in this case are also called ligand classification because the compounds are sorted out based on their ligand structure. So instead of having characteristics such as blue, red, or tall, the characteristics in this case would consist of a particular chemical structure such as a benzene ring as one category or a carbonyl carbon as another category

Step Three: Setup Training

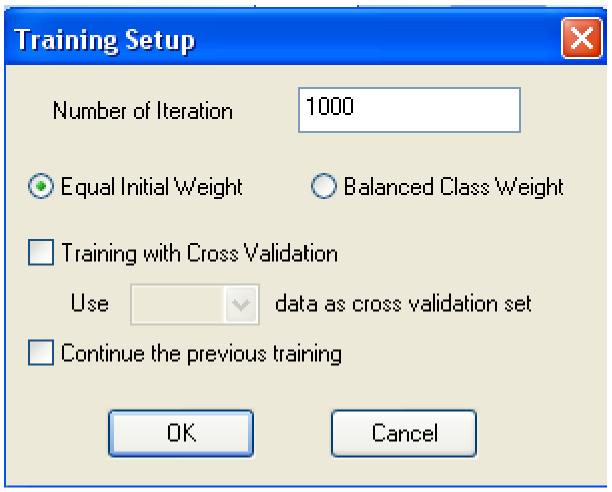
Once both categories of training compound datasets are imported, automatic datamining can be carried out to derive a predictive LiCABEDS model. This procedure is called model training or model development. In model training, FCC will examine the distinct patterns between two categories of compounds and build classifiers accordingly.

LiCABEDS is the imbedded machine-learning algorithm in FCC. LiCABEDS assumes an ensemble model for ligand classification:



A LiCABEDS classifier consists of a set of “decision stumps”, or so-called “weak learners”. Each decision stump inspects one feature and makes classification based on the feature. For example, the figure above illustrates the structure of a LiCABEDS model for agonist/antagonist classification. The final prediction from LiCABEDS depends on the voting of all the decision stumps with weight a_m. The model training in FCC just aims to derive such a classifier through constructing M decision stumps and assigning their weights.

To setup training, user can click ▶ on the toolbar, or go to “welcome page” and click “Setup Training”. Then, a dialog will pop up to collect a few parameters:


Training Setup

Number of Iteration: 1000

[✓ Equal Initial Weight]   Balanced Class Weight

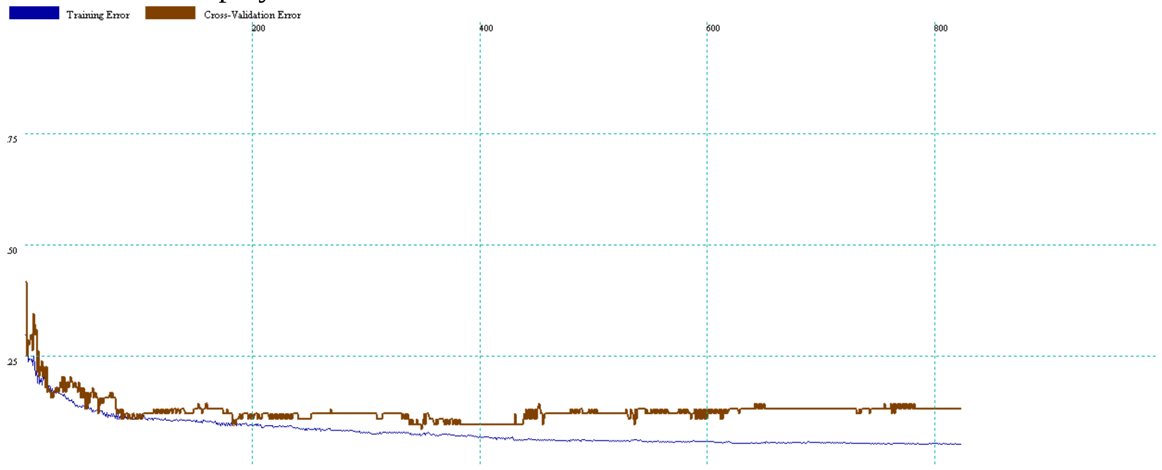
☐ Training with Cross Validation

Use ___ data as cross validation set

☐ Continue the previous training

The most important parameter in LiCABEDS training is the number of iteration, which is equal to M mentioned in the figure above. M determines the complexity of a LiCABEDS predictive model. A large M will generate more decision stumps to characterize the concern ligand properties, facing the risk of overfitting, while a small M will make a brief generalization with limited classification power. According to the test calculation, a large value of M usually yields decent prediction accuracy, e.g. M equal to the length of fingerprint. However, users still have the option to carry out cross-validation to search for the optimal value of M. In this case, check “Training with Cross Validation” and select the percentage of training datasets as cross-validation set. Users can also choose the initialization condition: equal initial weight or balanced class weight. If equal initial weight is chosen, then all the training compounds are treated equally. Nevertheless, this initialization condition is not suitable for many ligand classification tasks, e.g. virtual screening. In virtual screening, thousands of compounds are labeled as inactive but only a few are reported as active. The program can simply treat all the compounds as inactive to achieve nearly 100% accuracy. To solve this problem, balanced class weight assigns equal weight to each category instead of treating each training sample equally. Balanced class weight is encouraged if unbalanced training datasets are provided. Interruptible training is another competitive feature of LiCABEDS. Users can always continue with the previous training by checking “Continue the previous training” box. In this case, generalization error will be minimized based on the established model by adding more weighted decision stumps. For example, a model is developed using M = 1000. However, the model performance is not satisfactory on the cross-validation set or testing dataset. Users can simply continue the previous training by setting M = 1000 again. Then, the final model will contain totally 2000 decision stumps. This feature allows reusing the developed model to generate a new classifier instead of developing from the beginning.

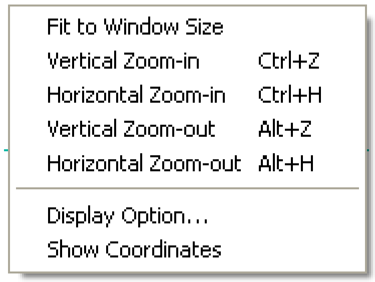
After clicking “OK” button, model training will start automatically and a real-time error curve will be displayed in the main window.



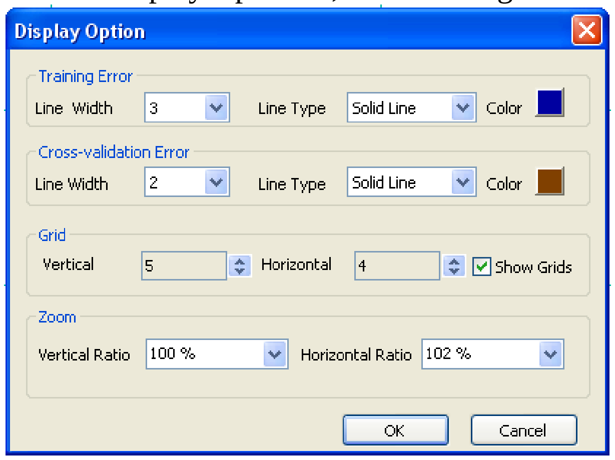
The screening shot displays undergoing model training. Certain functions are disabled during model training. Users can click 🟦 button in the tool bar to interrupt the training. The vertical grids indicate the number of decision stumps, and the horizontal grids indicate the training error or cross-validation error (measured by ratio). As time passes by, the training error curves move towards right until M steps have been finished. The legends in the upper-left explain the meaning of each curve. For example, the blue curve in this figure shows training error whereas the brown curve shows cross-validation error. In general, training error is minimized in a stepwise manner. As m (the number of finished iterations, m<M) increases, the training error usually decreases. However, the cross-validation error may not necessarily follow the same trend. The cross-validation (CV) error may forms a “U” shape or an “L” shape as a function of m. LiCABEDS also has the risk to overfit the training data as most supervised learning algorithms. Therefore, the CV error will reduce at the beginning when the training error is minimized, and then

increase due to overfitting, which forms a “U” shape error curve. By running cross-validation, users can pick up the optimal value of M according to the CV error. Each coordinate (x, y) denotes the percentage of mistakes, y, made on the training datasets or cross-validation dataset after x decision stumps have been included into the LiCABEDS model.

Program FCC has a default style to display training curves, but a custom display style can be set as well. Click right button in the window. Then a popup menu will show up:



Select “Display Option..”, then a dialog box will be displayed and collect users’ input.



Display Option

Training Error
Line Width 3 Line Type Solid Line Color

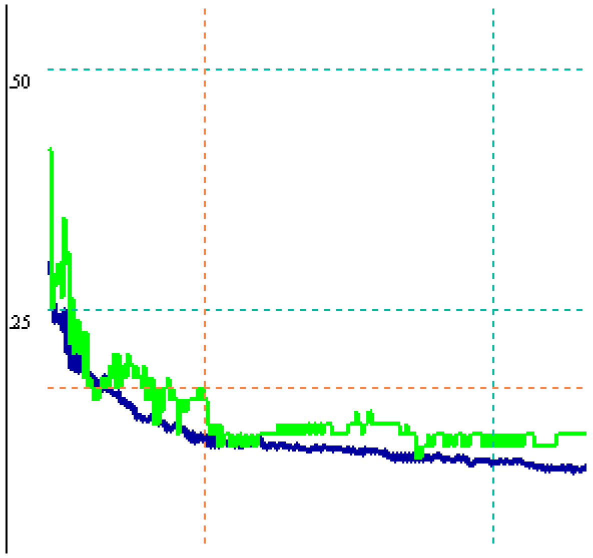
Cross-validation Error
Line Width 2 Line Type Solid Line Color

Grid
Vertical 5 Horizontal 4 [✓ Show Grids]

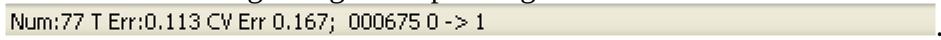
Zoom
Vertical Ratio 100 % Horizontal Ratio 102 %

First, users can choose the line width for either training error curve or cross-validation error curve in the unit of pixels. Then, users can specify the line color and style, such as dotted line and dashed line. The grid section controls the number of vertical and horizontal grids. Un-checking the “Show Grids” box will disable any assistant grids on the main window. Finally, set the zoom ratio through “Vertical Ratio” and “Horizontal Ratio” drop-boxes. After confirmation of the display style, the setting will become the default format for all the FCC program sessions.

To keep track of the detailed coordinate information, check “Show Coordinates” from the popup menu by clicking the right mouse button. Next, move the cursor in the window, FCC will automatically track the selected “weak learner” or decision stump, which is illustrated in the screenshot below:



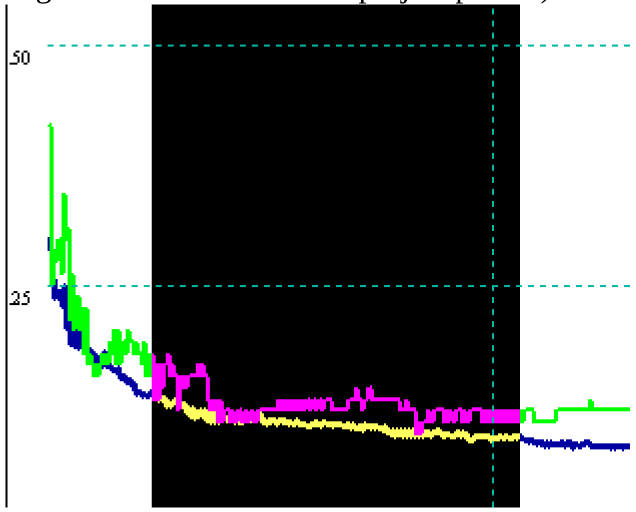
The information regarding corresponding “weak learner” is shown in the status bar:



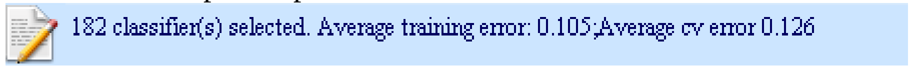
Num:77 T Err:0.113 CV Err 0.167; 000675 0 -> 1

.

In this example, the cursor is pointing to the 77th “weak learner” (Num: 77, zero-based index). Up to m = 78, the training error is 0.113, which means the LiCABEDS model can classify 88.7% of training samples correctly. As cross-validation is enabled in training mode, cross-validation error is also reported at the same time. In this case, the CV error is 0.167. The last part of the string, “000675 0 -> 1” indicates the hypothesis of 77th decision stump in the LiCABEDS model. This decision stump means that if the 675th bit (zero-based index) of fingerprint is 0, this compound is classified as positive (positive or negative class is defined in project profile).



FCC also supports range selection of “weak learners” by “dragging” the mouse with the left button down. The selected decision stumps have inverted color, and the summary will be given in the Message Board. All of these features help users to interpret the model and search for the optimal parameters.



182 classifier(s) selected. Average training error: 0.105;Average cv error 0.126

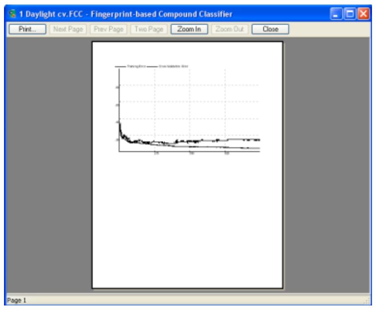
Tip: FCC can calculate the best display ratio and fit the training curve in the window. Go to View menu and select “Fit to Window Size”. If the size of the plotting is larger than the display window, the scroll bars will automatically show up. Shortcut keys are listed to scroll vertically or horizontally and zoom in/out:

Roll the mouse wheel to scroll horizontally
Roll the mouse wheel with “Shift” key down to scroll vertically
Vertical Zoom-in: Ctrl + Z
Vertical Zoom-out: Alt + Z
Horizontal Zoom-in: Ctrl + H
Horizontal Zoom-out: Alt + H

The selected of “weak learners” can be copied and pasted into any text-editing software, e.g. Wordpad. Select part of the model and click button and paste the content into any text editor:

```
id 158  bit 626==1->1   weight 0.124393 T Error 0.102607      CV Error 0.130952
id 159  bit 327==0->1   weight 0.122432 T Error 0.102607      CV Error 0.142857
id 160  bit 262==1->1   weight 0.119565 T Error 0.101766      CV Error 0.142857
id 161  bit 186==0->1   weight 0.125820 T Error 0.103448      CV Error 0.130952
id 162  bit 807==1->1   weight 0.115970 T Error 0.100925      CV Error 0.130952
id 163  bit 718==0->1   weight 0.124027 T Error 0.103448      CV Error 0.130952
id 164  bit 574==1->1   weight 0.122359 T Error 0.100084      CV Error 0.130952
id 165  bit 684==0->1   weight 0.120857 T Error 0.101766      CV Error 0.130952
id 166  bit 562==1->1   weight 0.114031 T Error 0.098402      CV Error 0.130952
id 167  bit 359==0->1   weight 0.122086 T Error 0.103448      CV Error 0.130952
id 168  bit 86==1->1    weight 0.119489 T Error 0.096720      CV Error 0.130952
id 169  bit 440==0->1   weight 0.119795 T Error 0.099243      CV Error 0.130952
id 170  bit 654==1->1   weight 0.099239 T Error 0.094197      CV Error 0.130952
```

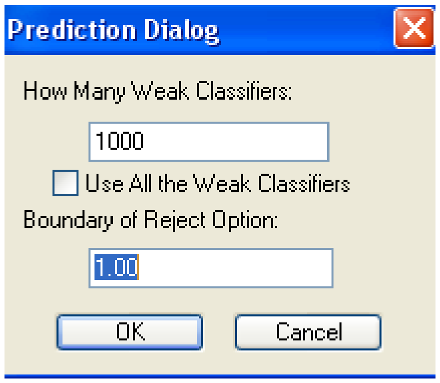
id field indicates the zero-based index of decision stump.
bit field tells the hypothesis in the corresponding decision stump. E.g. the 158th decision stump means if 626th bit is equal to 1, then the compound is classified as positive.
weight is equal to a_m in LiCABEDS model, which tells how much the decision stump contributes to the final prediction.
These fields are followed by T error (training error) and CV error (cross-validation error).
LiCABEDS error curve can be output to any installed physical or virtual Windows printer, like PDF printer. FCC supports what-you-see-what-you-get printing preview. Click “File -> Print Preview” and the printing preview will be shown in the program window:



The figure can be sent to a specific printer through a uniform printing dialog for archive or publication.

Step Four: Make Predictions

Once a predictive LiCABEDS model is developed and testing compound dataset is loaded, ligand classification can be carried out to predict ligand properties for the testing compounds. Click button on the toolbar, a prediction dialog will popup to collect parameters:

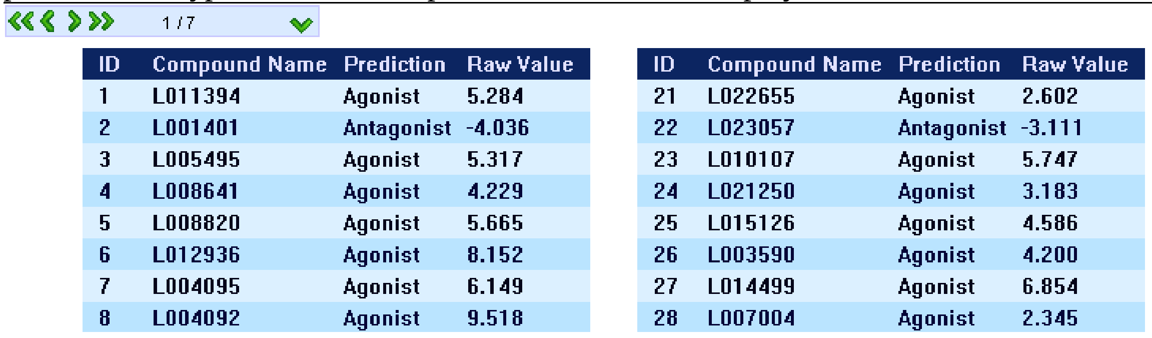


How Many Weak Classifiers: 1000

☐ Use All the Weak Classifiers

Boundary of Reject Option: 1.00

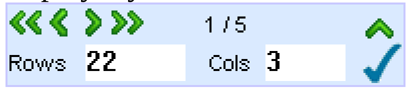
It is optional to use part of the decision stumps to make predictions, even if model is trained using a large value of M. For example, a LiCABEDS model is trained by 10000 iterations, only the first 1000 decision stumps may be used for predictions. If "Use All the Weak Classifiers" is checked, the corresponding edit box turns grey and the whole LiCABEDS model is used for prediction. Otherwise, users can input arbitrary number, as long as it is smaller than M. The other parameter in the dialog is the boundary of reject option. LiCABEDS can not only output the categorical value for each testing sample, but also output the degree of confidence for each prediction. The rationale and benefit are discussed in the manuscript on LiCABEDS. If the degree of belief is below the boundary value, LiCABEDS will simply output an "unknown" label instead of making a risky prediction. A typical view of the prediction window is displayed below:



ID	Compound Name	Prediction	Raw Value
1	L011394	Agonist	5.284
2	L001401	Antagonist	-4.036
3	L005495	Agonist	5.317
4	L008641	Agonist	4.229
5	L008820	Agonist	5.665
6	L012936	Agonist	8.152
7	L004095	Agonist	6.149
8	L004092	Agonist	9.518

ID	Compound Name	Prediction	Raw Value
21	L022655	Agonist	2.602
22	L023057	Antagonist	-3.111
23	L010107	Agonist	5.747
24	L021250	Agonist	3.183
25	L015126	Agonist	4.586
26	L003590	Agonist	4.200
27	L014499	Agonist	6.854
28	L007004	Agonist	2.345

At the same time, a prediction summary is also given in the message board. The toolbar at the top of the prediction page allows user to browse through the predictions and change display layout. Click the button to expand the toolbar:

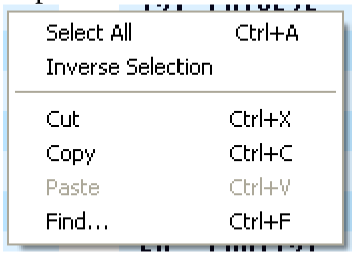


Rows 22 Cols 3

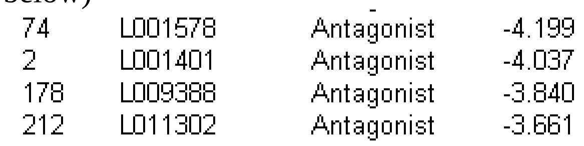
Functions for each button: "<<" : go to the first page; "<": go to the previous page; ">" go to the next page; ">>" go to the last page. These buttons are followed by the page number. Users can specify the number of rows and columns in each page. When the program starts, the number of rows and columns are specified automatically to fit content in one page. However, user can still change the display layout and click the "check" button to apply the changes.

In the prediction window, four data fields are displayed for each testing compound: compound ID in the testing dataset, compound name, predicted categorical value and the raw output value from LiCABEDS model. By default, all the testing compounds are ranked according to their indices in the testing dataset. However, they can still be sorted according to other fields. Move the mouse cursor to the first row of the table. The shape of the mouse cursor will be automatically changed to up-arrow or down-arrow. Click left mouse button to sort all the testing compounds according to the corresponding data field.

A testing compound entry can be selected by clicking left mouse button. "Ctrl" key needs to be pressed for selecting multiple items. User can open a popup menu by clicking right mouse button to select all the data (Ctrl + A) or inverse the selection. The "Cut" function can delete the selected testing data and put them into the clipboard (note: the data in the clipboard will be overwritten if another copy or cut request is made).



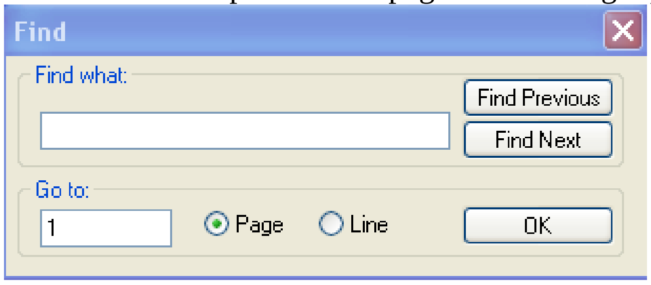
The selected testing data entries can be copied into the clipboard and pasted to texting edit software or a session of FCC program. The format copied to text editor still includes compound id, name, predicted category and raw prediction value (screenshot is shown below)



```
74      L001578     Antagonist    -4.199
2       L001401     Antagonist    -4.037
178     L009388     Antagonist    -3.840
212     L011302     Antagonist    -3.661
```

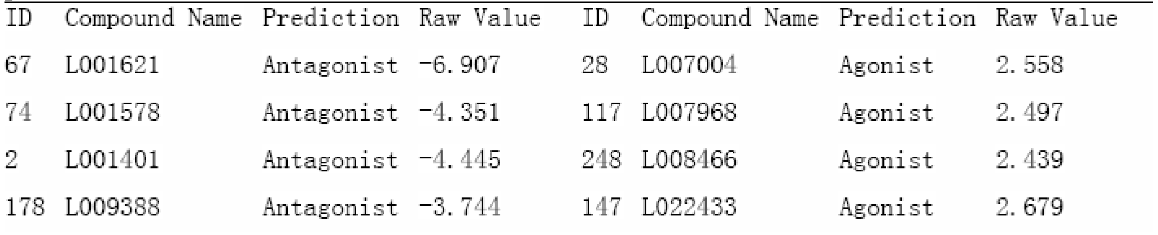
The data saved in the clipboard can be also pasted in to the same session of FCC or another session of FCC program. Nevertheless, different sessions of FCC may hold different predictive LiCABEDS models, so the old predicted category value is not necessarily consistent with the predictions from the new one. Predictions have to be made for the pasted data entries; otherwise, "unknown" labels will be assigned.

A "Find" dialog is provided to facilitate text searching. The interface of the dialog consists of text input box and page/line locating input box.



In the “Find what” textbox, input the exact or partial string of the compound name. Then click “Find Previous” or “Find Next” to carry out forward or backward searching. If no item is selected, the searching will start from the first entry in the table. Message board will report whether a matched compound name can be found. If so, the row is reported. The retrieved item will be automatically selected and labeled by red rectangle. It may be a good idea to jump straight to certain page or line instead of clicking “>” button many times, when large amount of testing data is imported into the program. Check “Page” or “Line” button and input the page or line number. After clicking “OK” button, the desired page or data entry will be automatically displayed.

Similar to the printing function in training module, the prediction results can also be sent to a virtual or physical printer. Once the paper size is specified, the program formats the layout and outputs the predictions line by line. However, printing preview function is not implemented because of the simple printing logic. The snapshot shows part of the PDF printout.

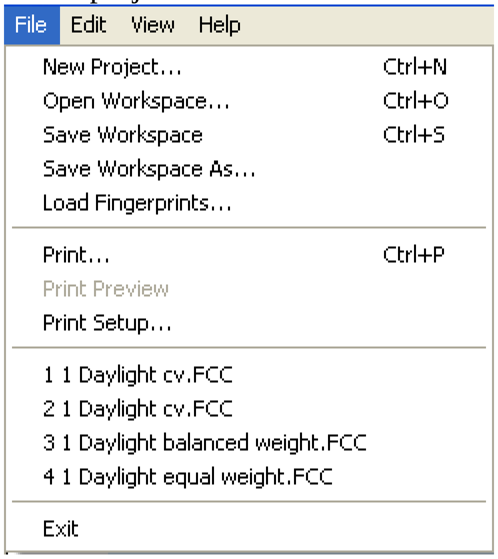


ID	Compound Name	Prediction	Raw Value	ID	Compound Name	Prediction	Raw Value
67	L001621	Antagonist	-6.907	28	L007004	Agonist	2.558
74	L001578	Antagonist	-4.351	117	L007968	Agonist	2.497
2	L001401	Antagonist	-4.445	248	L008466	Agonist	2.439
178	L009388	Antagonist	-3.744	147	L022433	Agonist	2.679

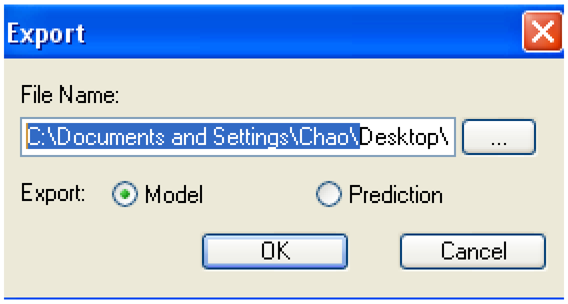
Save Your Workspace and Export Your Results

A project can be saved into a file, so that the project can be restored to the previous status. The saved workspace contains project profile, training and testing compound datasets, developed model and predictions. Before exiting the program, a dialog box will popup to remind saving the workspace if any changes has been made to the project. To save the project, click button on the toolbar. If the project is not associated with a workspace file, a file name needs to be specified in a file dialog in order to save the project. Otherwise, the project will automatically overwrite the associated file. To provide an alternative file name, go to "File" menu and select "Save Workspace As".

The project workspace file has a default "FCC" extension file. To restore the previous workspace, open another FCC program and drag a FCC workspace file into the main window. Another way to open a FCC workspace file is to click button and select a saved workspace file with extension .FCC. FCC automatically records the most recent workspace files and provides shortcuts for opening those files. User can get instant access to previous projects by expanding the "File" menu. The third section lists four most recent project files. Click the file name to open saved project.



FCC workspace format is a binary file format that can only be understood by the FCC program. However, an "export" function is implemented to enable exporting legible text results, such as predictive models and LiCABEDS predictions. To get access to this function, go to "Edit menu" and select "Export". A dialog box will follow up to gather more information:



File Name: C:\Documents and Settings\Chao\Desktop\

Export: [✓ Model] Prediction

Specify the file name by clicking "…" button and select to export predictive model or predictions. The exported model is in pure text format:

```
Totally 1100 weak classifiers:
ID      bit     value   weight
0       722     0       0.866193
1       54      0       0.491575
2       92      0       0.469724
3       641     0       0.434143
4       355     0       0.490002
```

Each row includes the index of a weak classifier, its hypothesis and its weight contributing to the final prediction (All the indices are zero-based). For example, the first weak classifier is "if the 722nd bit of fingerprint is equal to 0, the compound is classified as positive, which is weighted by 0.866193". The final prediction is the weighted summation of the outcome of each weak learner. If the summation is larger than 0, the output positive categorical value, otherwise, negative categorical value. Users may use the exported result to interpret the model and extract key features. Scripts can be also developed for web-based online prediction by applying the exported model. The format for exported predictions is similar to the clipboard text format that is mentioned in the previous section:

```
Totally 276 testing samples:
ID      name    class   raw value
1       L011394 Agonist 5.404472
2       L001401 Antagonist      -4.444728
3       L005495 Agonist 5.931770
4       L008641 Agonist 4.333997
5       L008820 Agonist 6.151025
```

Appendix

Molprint2D Fingerprint Generation

Molprint2D fingerprint characterizes a two-dimensional compound structure by a set of atom environment that is defined by Sybyl atom type. Reference regarding Molprint2D fingerprint is available at http://cheminformatics.org/molprint_download/. In our test calculation, Molprint2D outperforms FP2, Unity and MACCS fingerprint. Thus, a short tutorial is given on how to generate Molprint2D fingerprints and import the descriptors into FCC project.

First step, convert your compound structure file into mol2 format using Openbabel or other cheminformatics toolkit. Molprint2D recommends representing chemical structures using implicit hydrogen atoms. This conclusion is also agreed by our calculation. A FCC project requires three separate files: positive-category training data, negative-category training data and testing data (if available). Molprint2D software package is developed for Linux platform. Install the software on a Linux computer and use the following command to generate raw Molprint2D fingerprint:

```
mol22aefp input output
```

Input is your mol2 structure file and output is a file name to save the raw fingerprints. Generate the fingerprints for all the datasets.

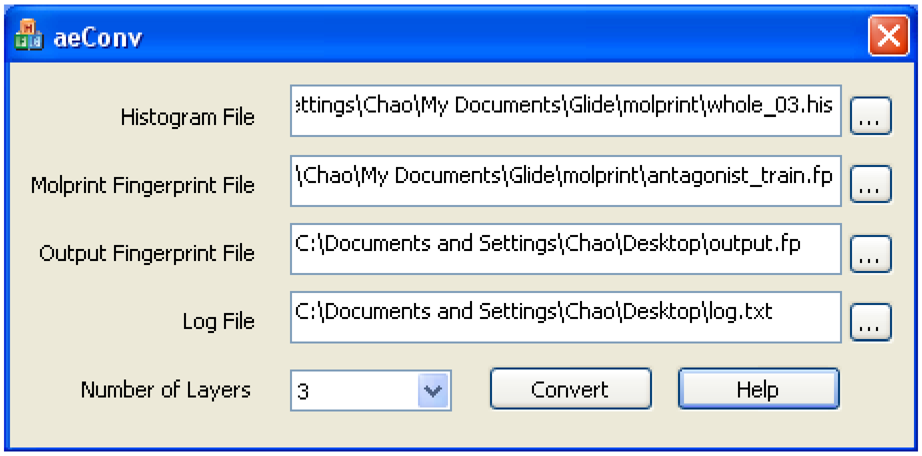
The raw Molprint2D fingerprint is in the following format:

```
L000007     4;0-3-0;1-4-0;2-4-0;2-1-2;    10;0-1-2;0-1-10;1-1-0;1-2-
2;2-2-0;2-2-2;    10;0-1-2;0-1-10;1-2-2;2-2-0;2-2-2;………
```

Each of the Molprint2D feature should be mapped to a unique index that indicates the presence or absence of the feature in a compound. To do this, a mapping list should be created. Luckily, Molprint2D package provides a command "build.pl" to generate a histogram of all the features. In this case, combine all of your structure data files into a single mol2 file and generate the raw Molprint2D fingerprint. Then,use the following command

```
build.pl input his-file –l min max
```

"input" is the raw fingerprint file for the whole dataset. His-file is the output histogram file name. The reason to combine all the compound datasets is that all the features in the project will be listed in the histogram and mapped to the descriptor vector accordingly. The last parameter "-l min max" means how many layers of atoms are considered in atom environment. It is usually set to "–l 0 2" for " –l 0 3". Finally, a variable selection can be carried out by modifying the output histogram file and deleting noisy features. Note that only features present in the histogram will be mapped to the descriptor. A feature will be discarded if it does not exist in the list.



Histogram File: ettings\Chao\My Documents\Glide\molprint\whole_03.his
Molprint Fingerprint File: \Chao\My Documents\Glide\molprint\antagonist_train.fp
Output Fingerprint File: C:\Documents and Settings\Chao\Desktop\output.fp
Log File: C:\Documents and Settings\Chao\Desktop\log.txt
Number of Layers: 3

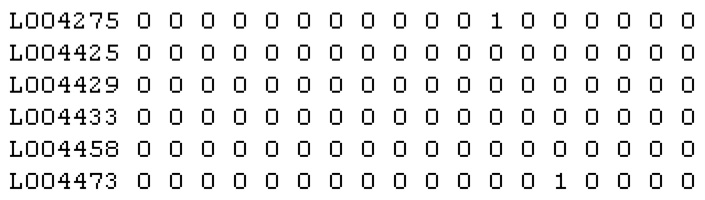
Download the Molprint2D converter from the LiCABEDS website. The screenshot of the program is displayed above. A fast string-matching algorithm is implemented in the converter using hashing table technique. It only takes a few seconds to process a megabyte data file. The program was initially developed as in-house script to facilitate Molprint2D conversion. Later, a graphical user interface was developed so that the program could be easily used by others. However, this program is not integrated into the FCC software and is not officially supported.

The converter requires five parameters including two input files, two output files and one fingerprint parameter. First, locate the histogram file by clicking adjacent "…" button and locating the histogram file that is generated from "build.pl" command. In the command "build.pl", users have an option to specify the number of layers that are considered in atom environment, such as "-l 0 2". "-l 0 2" means the atom environment contains atoms from neighboring atoms to atoms with two bonds away (more details can be found from official Molprint2D website). Choose the number of layers from the drop box. This value has to be consistent with the number of layers in the histogram file. For example, if no "-l" argument follows the "build.pl", the default maximum layer is 2 and the number of layer parameter should be set to 2. If a "-l min max" argument is specified in the command, then the number of layers in the Molprint2D converter should be equal to "max". The histogram file serves as a mapping table. Next, in the Molprint Fingerprint File edit box, locate the raw Molprint2D fingerprint file that will be converted into dichotomous vector format. Finally, input the "log file name" and fingerprint file name in which the converted fingerprint data will be recorded. Press "Convert" button to start data processing. Sample data files are available online:

Histogram.his: the preprocessed histogram file for the converter
Sample Molprint2D.fp: a raw Molprint2D fingerprint file

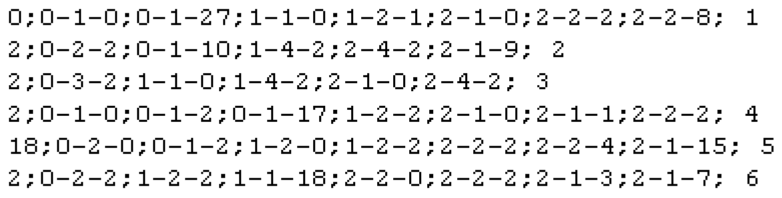
After applying the Molprint2D converter, the processed fingerprint in vector format and corresponding log file from the sample data files is given below:

Fingerprint file in vector format:



```
L004275 0 0 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0
L004425 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
L004429 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
L004433 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
L004458 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
L004473 0 0 0 0 0 0 0 0 0 0 0 0 0 1 0 0 0 0
```

The log file containing the whole mapping table:



```
0;0-1-0;0-1-27;1-1-0;1-2-1;2-1-0;2-2-2;2-2-8; 1
2;0-2-2;0-1-10;1-4-2;2-4-2;2-1-9; 2
2;0-3-2;1-1-0;1-4-2;2-1-0;2-4-2; 3
2;0-1-0;0-1-2;0-1-17;1-2-2;2-1-0;2-1-1;2-2-2; 4
18;0-2-0;0-1-2;1-2-0;1-2-2;2-2-2;2-2-4;2-1-15; 5
2;0-2-2;1-2-2;1-1-18;2-2-0;2-2-2;2-1-3;2-1-7; 6
```

Note: only part of the result is displayed

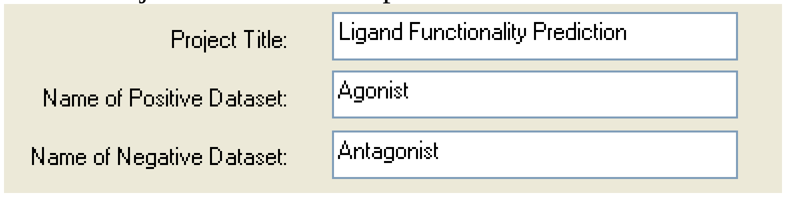
The fingerprint complies with the requirement of FCC format, so it can be imported to the program without any modification. The log file reveals the interpretation of the fingerprint vector. It tells the mapping mechanism of the fingerprint conversion. For example, feature “`2;0-2-2;0-1-10;1-4-2;2-4-2;2-1-9;`” is mapped to the second bit in the fingerprint vector. In other words, if the second bit of the fingerprint is 1, it indicates the presence of the feature in the compound; otherwise, the feature is absent. Therefore, users can interpret a predictive model and results according to the mapping schema.

Simple Walk-through Project

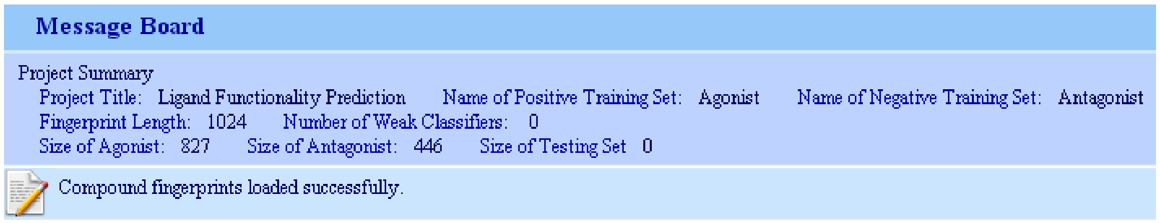
Sample training and testing datasets can be downloaded together with the program, FCC. The goal of the walk-through project is to model ligand functionality for 5HT1A subtype G-protein coupled receptor (GPCR). Ligands are classified into two categories: agonists that active the receptor and antagonists that inhibit or deactivate the receptor. In this experiment, a LiCABEDS model will be developed based on labeled agonists and antagonists. Then, the model will be used to make predictions on the testing compound datasets. The model performance is evaluated by comparing the predicted categorical value with the real labels. Molecular fingerprints have already been generated for all the compounds so that the datasets can be directly imported into FCC program. Here is a list of training and testing data files:

daylight_agonist_train.fp daylight_antagonist_train.fp The training agonist and antagonist datasets. These two datasets are used to develop a predictive model.
daylight_agonist_test.fp daylight_antagonist_test.fp The testing agonist and antagonist dataset. These two datasets are used to evaluate the performance of the developed LiCABEDS mode. All the compounds in the "daylight_agonist_test.fp" are labeled as agonists. Ideally, the program should output "agonist" for all the testing samples in the dataset. Similarly, all the testing samples in "daylight_antagonist_test.fp" are known to be antagonist.

Protocols:

1. Launch the "FCC.exe"
2. Click "Project Profile" and input the content as shown below:

Project Title: Ligand Functionality Prediction
Name of Positive Dataset: Agonist
Name of Negative Dataset: Antagonist
Click "OK" button.
3. Click "Load Fingerprints". Select "Agonist" option and locate file "daylight_agonist_train.fp" (click "…" button for browsing).
4. Select "Antagonist" option and locate file "daylight_antagonist_train.fp"

5. The Message Board should display the following information

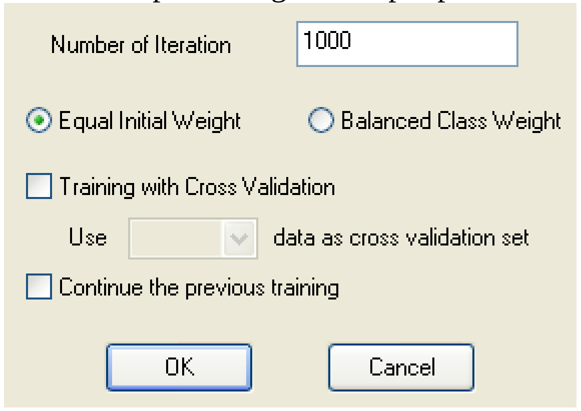


Message Board

Project Summary
Project Title: Ligand Functionality Prediction   Name of Positive Training Set: Agonist   Name of Negative Training Set: Antagonist
Fingerprint Length: 1024   Number of Weak Classifiers: 0
Size of Agonist: 827   Size of Antagonist: 446   Size of Testing Set 0

Compound fingerprints loaded successfully.

6. Click "Setup Training" and input parameters:



Number of Iteration: 1000

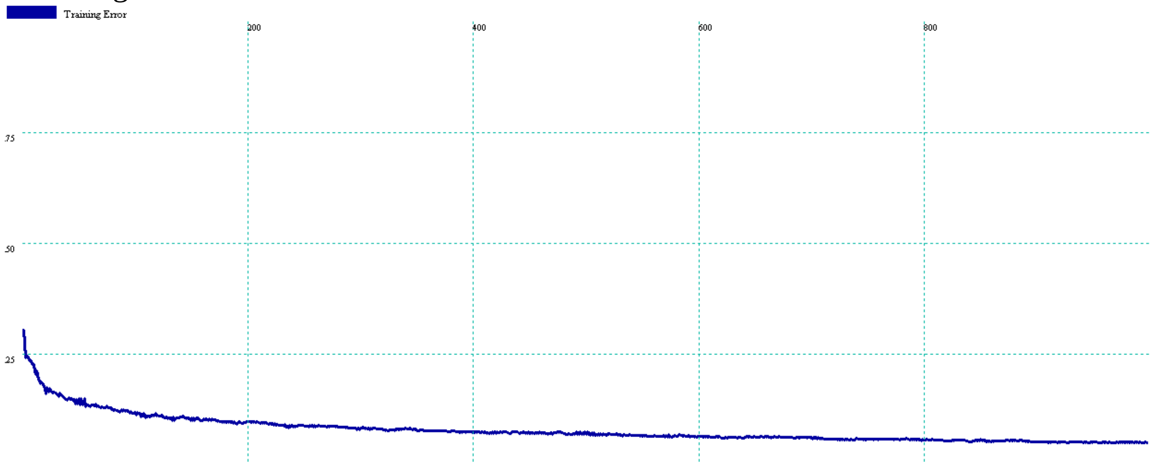
[✓ Equal Initial Weight]   Balanced Class Weight

☐ Training with Cross Validation

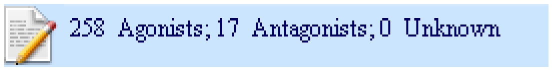
Use [ ] data as cross validation set

☐ Continue the previous training

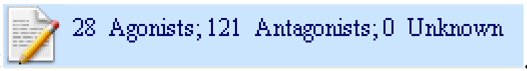
7. Click "OK" and training will start automatically. This process may take a few minutes. Please weight until the training curve reaches the right end of the window.
8. The error curve is minimized in a stepwise manner. The screenshot of the error curve is given below:



9. Load the testing dataset. Go to "File menu" and click "Load Fingerprints…". Select "testing dataset" and locate file "daylight_agonist_test.fp". Click "OK".
10. The main window will automatically display the home page. Click "Make Prediction". Take the default parameters and click "OK".
11. The predictions are listed in the main window, and a summary is given in the Message Board. The programs identified 258 compounds as agonists and 17 as antagonists. As we know, all the compounds are labeled as agonists. Therefore, the program makes 17 mistakes out of 275 samples.



```
258 Agonists; 17 Antagonists; 0 Unknown
```

12. Load the other testing dataset. Go to “File menu” and click “Load Fingerprints…”. Select “testing dataset” and check “Clean Existing Compounds in the Selected Dataset”. This is an important set to clean up the data imported before. Locate file “daylight_antagonist_test.fp”. Click “OK”.
13. Click “Make Predictions” and the result is summarized in the Message Board: 

```
28 Agonists; 121 Antagonists; 0 Unknown
```

. As all the testing compounds are labeled as antagonists, 28 mistakes are made by the program.

The overall prediction accuracy is about 89% (45 mistakes out of 424 testing samples)
Finally, press “Ctrl + S” to save your work session. The workspace will be restored to the previous status by loading the project file.