**KNIME workflow to assess PAINS filters in SMARTS format.**

**Authors…? Bael, Saubern, ++++**

Baell recently published a list of filters to remove problematic structures (PAINS) from screening campaigns. Has received considerable interest.

The filter list was published in Sybyl Line Notation (SLN) format, a format (only?) useable by the proprietary Sybyl software package.

Efforts by Guha to convert these SLN filters to the SMARTS format so that the filters can be used in a broader range of software packages were recently published.

As this conversion was a manual effort, concern has been expressed that these SMARTS filters will not deliver exactly the same structural matches as the original SLN filters, thereby including structures in a screening set known to fail the PAINS filters.

Additionally, different chemistry software packages might execute these SMARTS searches using different algorithms, again resulting in a different selection of molecules to the original SLN filters. Options used to import the target structure list, such as aromatization, desalting, or protonation, might also be the cause of different results.

Hence, we wished to make use of an open platform to test the SMARTS filters and benchmark different chemistry software packages in an intuitive manner. To this end, we chose the opensource and freely available Konstanz Information Miner (KNIME). This is a data analysis platform consisting of a GUI workflow, or 'pipeline', interface containing several chemistry related nodes. Workflow can be exported and distributed freely to other users and work across the three platforms currently supported by KNIME (Linux, MacOS, Windows). KNIME is distributed with the Chemistry Development Kit (CDK) and, recently, the RDKit (reference? www.rdkit.org) software packages. KNIME can access additional chemistry software packages by use of its 'external tool' node.

However, a disadvantage of KNIME is that only basic settings and functions are accessible for the included chemistry packages, as functionality has been sacrificed for ease of use. Further, the distributed packages are often not the most recently available versions and may contain bugs that have already been corrected in current versions. These shortcomings can be overcome by accessing recent versions of the packages using the built-in 'external tool' node, but introduces an extra level of complexity for the user.

Finally, KNIME workflows can be distributed with default input files. To standardize comparisons of the SLN and SMARTS filters, we have been making available a reference set of 10,000 structures from our internal library, and Guha has made the PAINS filters available in SMARTS format on his website. In the absence of a user selecting inputs, these files are used in our workflow.

The workflow appears in the supplemental information of this paper, and is also available for download at the www.myexperiment.org website. The user can execute the entire workflow simply by selecting the 'Execute All' menu item in KNIME.

workflow.tiff

Figure1: Overall structure of PAINS filter workflow using the RDKit package.

The PAINS workflow has two input paths. The target structure input path is shown in the upper half of Figure 1, and is set to read in the default reference set of 10,000 structures, mentioned above, in SMILES format. Alternative but currently unconnected nodes are shown for other input formats, including SD format, manual SMILES entry, or by direct extraction from a database. This last node will require additional expert configuration by the user.

The PAINS filter input path is shown in the lower half of Figure 1, and by default loads the SMARTS filters from Guha.

The workflow also has two outputs. The first is a file containing the 'good' structures and registration ids, plus any additional columns that the user chooses, as a tab separated text file. The user can easily add alternative output nodes to produce Microsoft Excel spreadsheets directly, or a standard SD file.

The second output file contains structures matching the PAINS filters. As a given structure may match more than one filter, we have used a 'GroupBy' node to concatenate multiple instances of each structure into one line. The output contains the name of the filter used in Ref 1, plus the SMARTS string itself that resulted in the match. This eases troubleshooting both the syntax of the SMARTS filters and errors present in the target structures.

**Results.**

The RDKit nodes contain no functionality to clean or normalize structures, but were successful in converting all 10,000 structures to its internal representation. Substructure searching using the PAINS SMARTS filters as queries produce 329 unique structural hits. The 'GroupBy' node could be temporarily modified to collate the results by filter name and frequency (Table 1). By comparison, the original SLN filters find 5xx unique structural matches.

Table 1: Number of times each PAINS rule matches a structure in the reference set (structures can match more than one rule) using the RDKit package.

|  |  |  |
| --- | --- | --- |
| rule name | count(RDKit) | count(SLN) |
| amino\_acridine\_A(46) | 8 |  |
| anil\_di\_alk\_C(246) | 17 |  |
| anil\_di\_alk\_D(198) | 17 |  |
| anil\_di\_alk\_E(186) | 20 |  |
| anil\_di\_alk\_furan\_B(2) | 7 |  |
| anthranil\_acid\_A(19) | 2 |  |
| azo\_A(324) | 38 |  |
| cyano\_ene\_amine\_A(56) | 11 |  |
| cyano\_imine\_B(17) | 4 |  |
| cyano\_keto\_A(2) | 1 |  |
| cyano\_pyridone\_A(54) | 7 |  |
| cyano\_pyridone\_C(11) | 1 |  |
| cyano\_pyridone\_D(5) | 1 |  |
| dhp\_amino\_CN\_B(9) | 2 |  |
| dyes5A(27) | 6 |  |
| ene\_cyano\_D(3) | 1 |  |
| ene\_five\_het\_C(85) | 9 |  |
| ene\_five\_het\_D(46) | 5 |  |
| ene\_five\_het\_H(6) | 1 |  |
| ene\_rhod\_A(235) | 18 |  |
| ene\_rhod\_C(13) | 1 |  |
| ene\_rhod\_F(8) | 1 |  |
| het\_565\_A(2) | 1 |  |
| het\_5\_D(2) | 1 |  |
| het\_65\_L(1) | 1 |  |
| het\_thio\_676\_A(10) | 2 |  |
| hzone\_acyl\_naphthol(22) | 2 |  |
| hzone\_anil\_di\_alk(35) | 3 |  |
| hzone\_enamin(30) | 2 |  |
| hzone\_phenol\_B(215) | 21 |  |
| hzone\_pipzn(79) | 11 |  |
| hzone\_pyrrol(64) | 6 |  |
| imine\_one\_A(321) | 31 |  |
| imine\_one\_fives(89) | 11 |  |
| imine\_one\_isatin(189) | 16 |  |
| keto\_phenone\_A(11) | 3 |  |
| mannich\_A(296) | 35 |  |
| styrene\_B(8) | 1 |  |
| sulfonamide\_A(43) | 5 |  |
| sulfonamide\_B(41) | 3 |  |
| sulfonamide\_E(2) | 1 |  |
| thio\_amide\_D(2) | 1 |  |
| thio\_urea\_C(9) | 2 |  |
| thio\_urea\_F(6) | 2 |  |
| thiophene\_amino\_Aa(45) | 6 |  |

In the original PAINS publication, the rules were divided into 3 separate filter blocks, and matching structures were removed from the target list before the remaining structures were submitted to the subsequent filter blocks. In order to compare directly the current work with this earlier work, we used the workflow to process the output file as the input from each of the three filter blocks (Table 2).

Table 2: Number of times each PAINS rule matches a structure in the reference set (structures can match more than one rule) using the RDKit package and the 3-step filtering process.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **block name** | **p\_m150** | **p\_l150** |  | **p\_l15** |  |
| **pains filter** | **count** | **pains filter** | **count** | **pains filter** | **count** |
| anil\_di\_alk\_C(246) | 17 | amino\_acridine\_A(46) | 8 | cyano\_keto\_A(2) | 1 |
| anil\_di\_alk\_D(198) | 17 | anthranil\_acid\_A(19) | 1 | cyano\_pyridone\_C(11) | 1 |
| anil\_di\_alk\_E(186) | 20 | cyano\_ene\_amine\_A(56) | 9 | cyano\_pyridone\_D(5) | 1 |
| azo\_A(324) | 38 | cyano\_imine\_B(17) | 2 | dhp\_amino\_CN\_B(9) | 2 |
| ene\_rhod\_A(235) | 18 | cyano\_pyridone\_A(54) | 7 | ene\_cyano\_D(3) | 1 |
| hzone\_phenol\_B(215) | 21 | dyes5A(27) | 6 | ene\_five\_het\_H(6) | 1 |
| imine\_one\_A(321) | 31 | ene\_five\_het\_C(85) | 8 | ene\_rhod\_C(13) | 1 |
| imine\_one\_isatin(189) | 16 | ene\_five\_het\_D(46) | 5 | ene\_rhod\_F(8) | 1 |
| mannich\_A(296) | 35 | hzone\_acyl\_naphthol(22) | 2 | het\_565\_A(2) | 1 |
| SUM | 213 | hzone\_anil\_di\_alk(35) | 3 | het\_5\_D(2) | 1 |
|  |  | hzone\_enamin(30) | 2 | het\_65\_L(1) | 1 |
|  |  | hzone\_pipzn(79) | 11 | het\_thio\_676\_A(10) | 2 |
|  |  | hzone\_pyrrol(64) | 6 | keto\_phenone\_A(11) | 3 |
|  |  | imine\_one\_fives(89) | 11 | styrene\_B(8) | 1 |
|  |  | sulfonamide\_A(43) | 5 | sulfonamide\_E(2) | 1 |
|  |  | sulfonamide\_B(41) | 3 | thio\_amide\_D(2) | 1 |
|  |  | thiophene\_amino\_Aa(45) | 6 | thio\_urea\_C(9) | 2 |
|  |  | SUM | 95 | thio\_urea\_F(6) | 2 |
|  |  |  |  | SUM | 24 |

------------

Still working on this bit. Doesn't look like it will work for now, as the CDK nodes only accept SMILES search strings and not SMARTS.

The CDK node in its default configuration fails to convert 164 of the SMILES strings in the reference set into its internal format. This is possibly due to the distribution of an outdated version within the KNIME package. Deselecting the 'Add explicit hydrogen' and 'Generate 2D coordinates' options only reduces the number of rejects to 138. Increasing the 'Process timeout' value from 10s to 50s resulted in 29 rejects and 100s gave 7 rejects. Increases above this value resulted in no improvement in the reject rate.

1. J. B. Baell and G. A. Holloway, *Journal of Medicinal Chemistry*, 2010, **53**, 2719-2749, http://dx.doi.org/10.1021/jm901137j.

2. C. Lipinski, in ACS 239, *MEDI 2*.

3. R. Guha. PAINS Substructure Filters as SMARTS. 2010-11-14. URL:http://blog.rguha.net/?p=850. Accessed: 2010-11-14. (Archived by WebCite® at http://www.webcitation.org/5uF4wHEpB)

4. M. R. Berthold, N. Cebron, F. Dill, T. R. Gabriel, T. Kötter, T. Meinl, P. Ohl, C. Sieb, K. Thiel, and B. Wiswedel, in *Studies in Classification, Data Analysis, and Knowledge Organization (GfKL 2007)*, Springer, 2007.

5. C. Steinbeck, Y. Han, S. Kuhn, O. Horlacher, E. Luttmann, and E. Willighagen, *Journal of Chemical Information and Computer Sciences*, 2003, **43**, 493-500, http://dx.doi.org/10.1021/ci025584y.

6. C. Steinbeck, C. Hoppe, S. Kuhn, M. Floris, R. Guha, and E. L. Willighagen, *Current Pharmaceutical Design*, 2006, **12**, 2111-2120, http://dx.doi.org/10.2174/138161206777585274.