PrePeP: A light-weight, extensible tool for predicting frequent hitters

Christophe Couronne, Maksim Koptelov, Albrecht Zimmermann

PrePeP: A light-weight, extensible tool for predicting frequent hitters*

Christophe Couronne, Maksim Koptelov, and Albrecht Zimmermann ✉

Normandie Univ, UNICAEN, ENSICAEN, CNRS – UMR GREYC, Caen, France
firstname.lastname@unicaen.fr

Abstract. We present PrePeP, a light-weight tool for predicting whether molecules are frequent hitters, and visually inspecting the subgraphs supporting this decision. PrePeP is contains three modules: a mining component, an encoding/predicting component, and a graphical interface, all of which are easily extensible.

1 Introduction

For more than a century, systematic drug development has led to a wide range of drugs for many illnesses and diseases. As a side-effect, much low-hanging fruit – easily identifiable components – has already been plucked. In addition, new pathogens arise, and known ones can develop resistances against existing drugs.

Modern drug development therefore involves *high-throughput screening* (HTS) [2], in which thousands or even millions of compounds are tested for activity against a given target. Since this remains a time-consuming process, HTS has been augmented by *virtual screening* [3], in which physical tests are replaced by predictions based on computational models.

In either process, the biggest hurdle are *false positives*, compounds that look like promising candidates during screening but turn into wasted time and money later on. This is annoying in the best of times but can have a much worse impact when time is of the essence, such as during the current Covid-19 pandemic, when large-scale efforts at underway to develop effective antiviral drugs.[1]

Among such false positives are so-called *frequent hitters* (FH), compounds that show activity in many assays, e.g. because they are non-specific *pan-assay interference compounds* (PAINS) [1]. While we addressed this problem in [4], the tool we presented there was an early prototype, and we have since improved it.

In particular, we have turned the tool more modular, breaking it down into:

- *The mining component*, which allows anyone with an sdf file containing frequent and non-frequent hitters to derive discriminative subgraphs.
- *The predictive component*, which, based on a training set and sets of subgraphs, predicts for molecules contained in an sdf file if they are FH or not.

* The tool can be downloaded at http://scientific-data-mining.org, "Software"

[1] https://lejournal.cnrs.fr/articles/covid-19-15-milliard-de-molecules-passees-au-criblage-virtuel

– *The graphical interface*, which allows for each molecule predicted FH to visualize the subgraphs that support the prediction.

There exist other tools for FH prediction [6, 5] but only in the form of web services. Running predictions for more than a few tens of molecules can take several hours during which the user gets no feedback on the process. Our tool, on the other hand, can be run locally, on as many machines as the user has available. It is easily extensible by users, is light-weight (2.9 MB in archived form), and depends only on a few widely available Python libraries.

2 Mining discriminative subgraphs

To explain the usage of the mining component, we quickly repeat the basics of the method proposed in [4]. A vital aspect of it is the mining of subgraphs discriminating between frequent and non-frequent hitters. Since frequent hitters are very much in the minority in most data sets, non-frequent hitters are subsampled to create balanced molecular data sets. This sampling is repeated to capture all information contained in the non-frequent hitters.

Structural information stored in sdf files is translated into gsp files, from which discriminating subgraphs are mined using a supervised *gSpan* implementation developed by Siegfried Nijssen. These subgraphs, finally, are used to encode the underlying data in terms of their presence and absence.

The code is written in Python and requires the networkx library. Launched on the command line, it takes as parameters the original sdf file and the number of subgraphs to be mined per sample. Its output is a folder containing the subgraph files as well as the encoded training data. For the classifier reported on in [4], based on $\sim$ 150k molecules, 200 samples, and 100 subgraphs per sample, the zipped folder amounts to 2.8 MB, easily transferable. This is of particular interest because neither the definition of frequent hitters nor the data sets to build models from are clear. Researchers can therefore easily experiment with different options.

3 Predicting frequent hitters

Prediction is done by learning a decision tree on each sample and taking the majority vote of those trees for a molecule to be predicted. Given the results reported in [4], we refrain from using molecular descriptors and base predictions solely on the mined discriminative subgraphs.

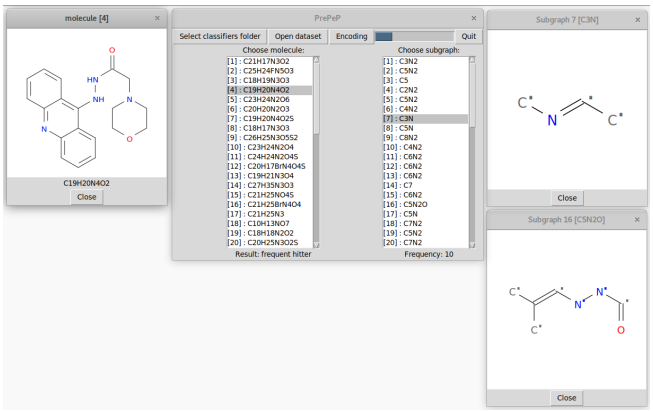
Also written in Python, and making use of openbabel 3.0.0, the module includes by default the data and subgraphs used in our earlier publication. A folder containing different data and subgraphs can be passed as a parameter on the command line. The name of the sdf file containing the molecules to be predicted is passed as a mandatory parameter. The module learns the appropriate number of decision trees, encodes the molecules to be predicted using the available subgraphs, and predicts for each molecule whether it is a frequent hitter or not.

Separating this component from the graphical interface described in the next section is a conscious decision. If the two were tightly integrated, it would be impossible to launch predictions remotely on a high-performance server to leave them running unattended for a while.

4 Visualizing graphs supporting the decision

While the predictive component would be the go-to option when working with a large data set, the biggest problem with PAINS is that the biochemical mechanism is in many cases not well understood. To support understanding the predictions, particularly in the case of molecules where the expert is not convinced by the prediction, the graphical interface *wraps* the predictive component.

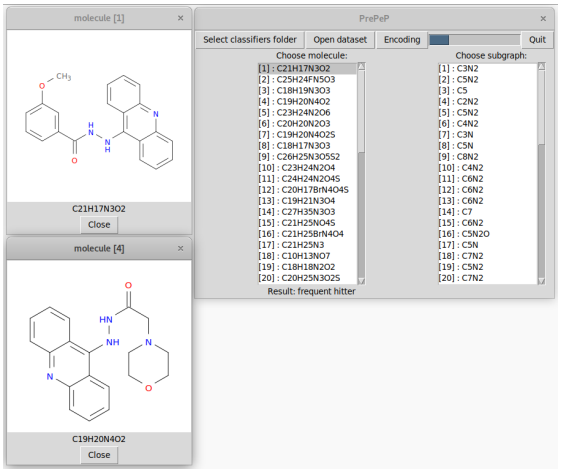
The interface supports the operations of the predictive component: loading data and subgraphs, loading molecules to be predicted, encoding those molecules (prediction is done automatically). Additionally, however, the user can select individual molecules (listed on the left) and if they have been predicted as being frequent hitters, the subgraphs supporting this decision are shown in the right.



As described above, prediction is performed by majority vote of a number of decision trees and we consider subgraphs to *support the decision* if they

- occur in inner nodes of decision trees predicting the molecule as FH, and
- the test whether they are present in the molecule is *true*.

Clicking on the molecule opens a separate window showing a visualization. Clicking on subgraphs, in turn, also open separate windows visualizing them, which allows to see where in the molecule the subgraph occurs, and (given the necessary expertise) whether basing prediction on the subgraph makes sense. Subgraphs are ordered by how often they were involved in the decision, with that information given at the bottom of the window for the selected subgraph.



In the same manner as for subgraphs, the user can select several molecules to compare visually. For the sake of presentation, we have grouped windows beside each other but all windows can be freely moved around, the help grouping graphs that one wants to consider together, for example.

5 Conclusion

We consider PrePeP a useful tool for chemoinformaticians who want to do quick-shot frequent hitter prediction. It can be easily deployed, multiple instances can be run on high-performance servers, and it can be extended with new/additional data and subgraphs.

References

1. Aldrich, C., Bertozzi, C., Georg, G.I., Kiessling, L., Lindsley, C., Liotta, D., Merz Jr, K.M., Schepartz, A., Wang, S.: The ecstasy and agony of assay interference compounds (2017)
2. An, W.F., Tolliday, N.: Cell-based assays for high-throughput screening. Molecular biotechnology **45**(2), 180–186 (2010)
3. Bajorath, J.: Integration of virtual and high-throughput screening. Nature Reviews Drug Discovery **1**(11), 882–894 (2002)
4. Koptelov, M., Zimmermann, A., Bonnet, P., Bureau, R., Crémilleux, B.: Prepep: A tool for the identification and characterization of pan assay interference compounds. In: Guo, Y., Farooq, F. (eds.) KDD. pp. 462–471. ACM (2018)
5. Matlock, M.K., Hughes, T.B., Dahlin, J.L., Swamidass, S.J.: Modeling small-molecule reactivity identifies promiscuous bioactive compounds. J. of chem. information and modeling **58**(8), 1483–1500 (2018)
6. Stork, C., Chen, Y., Šícho, M., Kirchmair, J.: Hit dexter 2.0: machine-learning models for the prediction of frequent hitters. J. of chem. information and modeling **59**(3), 1030–1043 (2019)