Vol. 15, No. 4, 2024
ISSN 2233-4203/ e-ISSN 2093-8950
ARTICLE www.msletters.org | Mass Spectrometry Letters

EI2FP: Efficient Prediction of Molecular Fingerprints from Electron Ionization Mass Spectra

Mikhail D. Khrisanfov[1,2]*, Dmitriy D. Matyushin[2], Andrey S. Samokhin[1,2], and Aleksey K. Buryak[2]

[1]*Department of Chemistry, Lomonosov Moscow State University, Moscow, Russia*
[2]*A.N. Frumkin Institute of Physical Chemistry and Electrochemistry, Russian Academy of Sciences, Moscow, Russia*

Received August 15, 2024, Revised November 5, 2024, Accepted November 7, 2024
First published on the web December 31, 2024; DOI: 10.5478/MSL.2024.15.4.178

Abstract : Obtaining information about the molecular structure from the mass spectra is one of the most pursued challenges in non-targeted analysis. The complete solution to the problem has not been found yet, therefore only partial information about the structure can be obtained from mass spectra, often in the form of various molecular fingerprints. One of the latest approaches for prediction of molecular fingerprints from electron ionization mass spectra is DeepEI, which suggested a suboptimal procedure based on using a separate neural network for each molecular fingerprint (more than 100 models in our work and 636 using the DeepEI method). More than that, after repeating the procedure described in the original article, we assumed that at least some of their models were most likely overfitted. We streamlined the original approach by predicting multiple types of molecular fingerprints with a single multi-output neural network. We developed a lightweight and performant architecture (called Lite model for brevity) with improved accuracy (0.91 vs 0.89), precision (0.86 vs 0.77), and recall (0.71 vs 0.70) compared to the DeepEI approach. Additionally, the Lite version of the model was more than 100 times faster than the DeepEI approach in training and inference.

Keywords : molecular fingerprints, electron ionization mass spectra, machine learning, deep learning

Introduction

The objective of obtaining molecular structure from mass spectra is one of the most sought in chemoinformatics and non-targeted analysis. However, there is still no full solution to this problem i.e. getting a complete structural formula from just an electron ionization (EI) or tandem (MS/MS) mass spectrum. Mass spectra of unknown compounds can be searched against a library of known compounds, but often the correct answer can be absent in the returned list because the number of existing chemical compounds is orders of magnitude more than the number of records in largest libraries.[1] There are also some approaches that allow generating *in-silico* mass spectra,[2-4] but the simulations of complex processes occurring during fragmentation are far from perfect. Therefore, the ability to estimate at least some parts of molecular structure (either functional groups, or some combinations of atoms) from mass spectra is quite beneficial. Molecular fingerprints (MFs) are computer-readable representations of some parts or functional groups of the molecule and are widely used for this task.

MFs can be calculated from molecular structure using a wide number of libraries[5-7] and standalone programs.[8-10] MFs can be used to compare chemical structures between each other and search for similar structures in databases.

MFs are also widely used as inputs for machine and deep learning models to predict physico-chemical properties of substances, for example, Kovats retention indices,[11] EI mass spectra,[12] retention times in reversed-phase liquid chromatography,[13] etc.

Some of the most common fingerprint types are: CDK,[6] PubChem,[14] Klekota-Roth,[15] MACCS keys,[16] circular extended connectivity fingerprints (ECFP, also called Morgan fingerprints)[17,18] and functional-class fingerprints (FCFP),[17] path-based Daylight fingerprints.[19] PubChem and Klekota-Roth fingerprints, as well as MACCS keys have a defined length of 4860, 881, and 166, respectively. More than that, each of these fingerprints describes a pre-defined structure or a group of atoms. At the same time hash-based fingerprints like ECFP, CDK, and FCFP can have arbitrary length, therefore they do not have a pre-defined meaning. In most cases 1024-bit hash-based fingerprints are used. A quick

*Reprint requests to Mikhail D. Khrisanfov
https://orcid.org/0000-0002-2364-6168
E-mail: khrisanfovmike@gmail.com

Google Scholar search seems to suggest that ECFP fingerprints are used significantly more than any other, with PubChem fingerprints taking the second place, and MACCS keys coming in third place by popularity.

There are several approaches for predicting fingerprints from mass spectra. CSI FingerID[20] uses a fragmentation tree to explain the experimental MS/MS spectrum of an unknown molecule and predict its molecular structure fingerprint. MetExpert[21] uses partial least-squares discriminant analysis for predicting 86 molecular substructures from an EI mass spectrum.

DeepEI[22] has a unique approach: each MF is predicted from an EI mass spectrum by a separate fully-connected neural network. Firstly, the authors of the DeepEI approach generated 8034 fingerprint bits (including CDK-standard, PubChem, Klekota-Roth, MACCS keys, Estate, and circular fingerprints) and selected 636 that showed a balanced representation (occurring between 10% and 90%). Secondly, to predict the whole set of MFs 636 separate neural networks were constructed; about 6.5 million weights were trained per model and stored for future use, resulting in a total of over 4 billion weights. Finally, MFs predicted with these models were used to improve the library search results for compounds absent in the library.

While the idea of predicting fingerprints from mass spectra suggested in the original DeepEI article is very popular,[23-26] most authors did not use it in their experiments directly. Some of them were inspired by DeepEI and built their own tools and neural networks[27] which could be more specialized[28] or use other types of spectra.[29,30] Other researchers noted that they are considering using DeepEI in their future works.[31] We can speculate that the practitioners that would benefit the most from using DeepEI were discouraged by the cumbersome process of training a multitude of networks to incorporate DeepEI into their existing pipelines.

More modern approaches are focused mainly on using MS/MS spectra as input data and high-resolution mass-spectrometry databases as training datasets. MSNovelist[32] predicts both MFs and structural information from MS/MS spectra. Mass2SMILES[33] also predicts structures from MS/MS spectra. IDSL_MINT[34] uses state-of-the-art transformer architecture to predict molecular fingerprints from MS/MS spectra.

The task of predicting fingerprints or structural information from EI mass spectra seems to be a bit forgotten as recent articles mainly focus on MS/MS (both high and low resolution) spectra[35], which contain more information. However, the task is far from being fully solved as the quality of prediction is nowhere close to ideal. As it was already mentioned, DeepEI approach[22] is suboptimal because each of the MFs is predicted by a separate neural network. Both training and inference speed are therefore severely limited and managing hundreds of models is tedious (see Results and Discussion).

We aimed to streamline the approach suggested by Ji *et al.* with DeepEI[22] by improving both accuracy, precision, recall as well as performance of the model. To keep up with modern trends in untargeted gas chromatography mass-spectrometry (GC/MS) analysis, the model needed to be lightweight to predict MFs for a large number of potential candidates almost in real-time (online). In this study we present a lightweight and performant Lite multi-output model that can predict ECFP fingerprints and MACCS keys from EI mass spectra more accurately and precisely than the DeepEI approach. The Lite model is more than 100 times faster than the DeepEI approach both in GPU training and CPU inference.

Experimental

Hardware

The following computer hardware was used in this work: CPU - Intel Xeon E5-2667 V2; RAM - 64 Gb DDR3 ECC REG @1600 MHz; GPU - Nvidia RTX 3060 12 Gb.

Software and libraries

Python 3.12 with PyTorch,[36] RDKit,[5] NumPy,[37] pandas,[38] matplotlib[39] and seaborn[40] were used to carry out this investigation. Binary Cross Entropy (BCE, nn.BCELoss) loss function was used for training all the neural networks. Jupyter Notebooks with code and supplementary materials are available on GitHub.[41]

"Mainlib" sub-library of the NIST 17 mass spectral database was used for training and testing all the neural networks. It was randomly split in a 9:1 ratio for training-validation and test datasets. The training-validation set was then randomly split in 85:15 for training and validation purposes. The validation dataset was used to optimize the architecture and hyperparameters of the networks and the test set was used only once - to compare the final metrics between our neural network and the DeepEI ones.

Morgan fingerprints (or extended circular fingerprints - ECFP) in the boolean variant with *radius = 3,* which equals ECFP6, were limited to 1024 bit length and calculated using rdkit.Chem.AllChem.GetHashedMorganFingerprint. These fingerprints were calculated for the whole dataset. In accordance with the DeepEI paper, 62 fingerprints with True value appearing in 10 to 90% cases were chosen as a subset to compare our suggested models against the DeepEI ones.

All 167 MACCS keys (available in RDKit) were calculated for the whole dataset. The same selection procedure resulted in 95 MACCS keys.

Models

DeepEI models

The DeepEI models were recreated with 4 layers (Figure 1): 2000 neuron input, 1000 and 500 neurons hidden layers and a sigmoid function as output layer (2000->1000->500->1). The activation function for the output layer - SoftMax from the original work[22] was unnecessary for the one-bit (True/False) classification task, therefore it was replaced

Figure 1. The architectures of the neural networks: DeepEI[22] with nn.Sigmoid activation function in the output layer, Full and Lite architectures suggested in this work.

with sigmoid activation function (nn.Sigmoid). The hyperparameters for the neural networks were the same as in the original publication: 10^{-3} learning rate, 32 batch, 8 epochs. As it was described in the original work, a separate DeepEI neural network was trained to predict each of the selected ECFP6 fingerprints and MACCS entries (65 and 92 respectively, see results and discussion).

Full model

The Full version of our architecture (Figure 1) had 4 linear/fully connected layers in the main body and 1 linear layer per each group of output layers: 1024 and 167 output neurons for ECFP6 and MACCS, respectively. The groups of output layers can be modified to predict other MFs. Our suggested models used SiLU activation function because it enabled smoother validation loss curves with better final results compared to the ReLU and Leaky ReLU. Dropout and BatchNorm layers were used in the main body to prevent early overfitting. The activation function in the output layer was sigmoid (nn.Sigmoid).

Lite model

The Lite version of the model had 3 linear layers in the main body and 1 linear layer per each group of output layers, SiLU activation functions, Dropout, and BatchNorm layers (see Fig. 1). There were two output layers: one with 1024 output neurons for fingerprints and the other with 167 neurons - for MACCS, both had a sigmoid (nn.Sigmoid) activation function.

AdamW optimizer with learning rate of 10^{-3} and 512 batch size was used for both Lite and Full versions of the architecture. Learning rates in range from 10^{-4} to 10^{-2} were tested with $\sqrt{10}$ multiplier increment.

Results and Discussion

Selecting fingerprints and inconsistencies in the DeepEI article, code and supplementary data

We chose ECFP6 fingerprints and MACCS keys for comparison of performance of our models against the DeepEI ones for several reasons: (1) these two types of finger-

prints were used in the original publication;[22] (2) they are widely used in the literature; (3) they represent two distinctly different types (hashed and pre-defined fingerprints); (4) they are accessible for calculation using RDKit. We found some inconsistencies in the DeepEI paper when we were trying to replicate their approach for selection of the fingerprints (10 to 90% of positives in NIST). The number of selected MACCS keys mentioned in the main text (44) significantly differed from the information presented in the supplementary materials (93). We selected 95 MACCS keys (10 to 90% of positives in NIST); this number was reasonably close to the number from the supplementary materials. We encountered the same problem with circular fingerprints (ECFP6): 67 mentioned in the article and only 61 in the supplementary files. We chose 62 ECFP6 fingerprints (10 to 90% of positives in NIST).

Dealing with overfitting

We carefully implemented and tested the original DeepEI approach according to the description provided.[22] Comparing the loss for validation and training datasets we can conclude that all the DeepEI neural networks (both for ECFP6 and MACCS) were severely overfitted. Each of the models achieved optimal validation loss after approximately 4 epochs and then it began to increase.

The variants of the fully-connected architecture without any regularization that were also tried in this work (Dropout or BatchNorm) displayed this overfitting behavior as well. The overfitting could be corrected without penalty in validation loss only by using some form of regularization. Both using several Dropout layers and combining them with BatchNorm allowed the neural networks to be much deeper and achieve better validation loss.

Advantages of SiLU activation function

Initial models in this work used ReLU[42] activation function as it is *de facto* a default one for fully connected and convolutional neural networks. However, while tuning the architecture of the models several other activation functions were tested: Leaky ReLU,[43] GELU[44] and SiLU.[45,46] Each of these activation functions was thought to improve the training qualities of the models. Leaky ReLU was introduced by its creators to solve problems with dying gradients during backpropagation; it added a negative Y part to the ReLU for X values less than 0. We did not encounter any difficulties with gradients, therefore we saw no measurable improvement either in validation loss or in training speed (number of epochs to fully train the model). GELU added stochastic behavior and dropout properties to the ReLU activation function via multiplication of input values by the cumulative Gaussian distribution.

$$GELU(x) = x\Phi(x) = x \times 0.5(1 + erf(x/\sqrt{2}))$$

Where *erf* is the Gauss error function:

$$\mathrm{erf}(z) = \frac{2}{\sqrt{\pi}} \int_0^z e^{-t^2} dt$$

The authors of the original publication[44] attribute enhanced performance of models with GELU activation function to its increased curvature and non-monotonicity compared to ReLU that allows better approximation of more complex functions. GELU was used in several intermediate variants of the architecture for the Full version, but was replaced by Sigmoid Linear Unit (SiLU).

SiLU (also known as Swish) was introduced in the same article as GELU[44] and was described as a simpler and faster, even if a bit worse, alternative. SiLU multiplies input values (x) by the sigmoid function applied to x:

$$SiLU(x) = x \times \sigma(x) = x/(1 + e^{-x})$$

As it was shown by Ramahadran et al.,[45] simply replacing ReLU activation function with SiLU significantly improved results of deeper models. It was suggested that the advantages of SiLU stem from its properties. Being unbounded above (x>>1) allowed SiLU to avoid saturation and slowness due to near-zero gradients; being bounded below (x<<0) applied strong regularization effects on the neural network. Non-monotonicity of SiLU improved gradient flow compared to ReLU, and its smoothness was beneficial for optimization and generalization of the neural network.[45] Even for the comparatively shallow Lite version of the network we saw the advantages of SiLU. The validation loss was lower than with ReLU and the values of the training and validation losses were closer to each other, which indicates better generalization properties of the model. Both the training and validation loss curves were smoother with SiLU compared to ReLU.

Choosing architecture of the neural networks

As it was mentioned in the Introduction, in the DeepEI approach[22] each of the MFs is predicted by a separate neural network. The performance is therefore limited and the management is unnecessarily complex. Our end goal was to get better classification metrics than DeepEI but streamline the approach by constructing only one neural network to predict all ECFP6 and MACCS keys simultaneously. The resulting model should preferably be as light and fast as possible to be included into a more complex GC/MS suite that can run on an average laboratory PC using only CPU without GPU acceleration. The model would be used to predict MFs from experimental EI mass spectra of unknown compounds in untargeted GC/MS analysis. These MFs would then be compared against the MFs calculated for the known molecules and used in the ranking algorithm. This approach would help eliminate some of the candidates and, potentially, help to elucidate molecular structure.

There were two directions that were explored using fully

connected architectures. Firstly, the architecture was tuned to achieve the lowest validation losses and highest validation metrics. The result of this stage was developing the so-called Full model. Secondly, there was a search for a slimmed down architecture that can achieve comparably low validation loss with a significantly smaller number of neurons. This search resulted in the Lite model, which represents the most lightweight architecture of the neural network and, as will be shown later, it is (almost) as good as the Full one.

Using convolutional layers is an effective way to lower the number of weights without sacrificing the performance. This approach seemed to be especially handy for obtaining a Lite model considering that mass spectra generally have clusters of peaks that do not need to be combined with all other *m/z* values (as when using dense layers), just *m/z* values inside the cluster and clusters between each other (as with deep convolutional neural networks - CNNs). However, preliminary testing showed that CNNs with lower number of weights produced higher validation loss than their fully connected counterparts, and CNNs with about the same number of weights were not faster than fully-connected models. Therefore from this point we focused on fully connected neural networks.

The search for a fully-connected lightweight architecture was conducted in two ways: firstly, starting from the full version of the model, the number of hidden layers was grown with simultaneous reduction in the number of neurons in each of the layers. Secondly, the number of layers was reduced with adding weights to each of the remaining layers. Then the number of neurons in the layers of the model chosen during the previous step was also lessened until the performance tradeoff was noticeable. This way, both the more shallow (which sometimes produce unexpectedly great results[47]) and the deeper architectures were investigated. In the end, the Lite model trades 9% higher BCELoss on validation dataset compared to the Full model for being approximately 3.5 times faster than the Full model.

Comparison of DeepEI approach against the suggested models

While the DeepEI approach shows decent performance in terms of metrics, there is a major drawback in methodology when each MF/MACCS is predicted by a separate model. Firstly, managing more than 100 models (65 ECFP6 and 92 MACCS values) is quite problematic. In the original approach, weights were loaded for each of the models one by one, which was unnecessarily complicated. Despite the relatively small size of all DeepEI models (3.8 GiB total), it is extremely inefficient compared to using concatenated input one time as in all our suggested architectures.

Secondly, dividing the MFs between the models may give each of them more neurons, but it disables the possibility of "cross-communication" where some of the base neurons close to the input layer are transforming the mass spectrum into information about a molecule. These neurons can be reused by several MFs and the information that comes to these neurons through backpropagation can boost the accuracy of prediction of other MFs, too.

Finally, all of the above along with using batch size of 32 made the training process very slow even with GPU acceleration: about 10 hours for 65 ECFP6 + 92 MACCS (only 8 epochs per model). In comparison, our Lite and Full models were trained on GPU for 6 and 20 minutes respectively (100 epochs). Inference of DeepEI models on CPU was also slow. Processing test part of the dataset (about 27,000 molecules) using DeepEI approach required about 3 and 5 minutes for calculating 62 ECFP6 and 95 MACCS respectively. Our Lite and Full models were 130 and 34 times faster (Table 1).

Comparison of accuracy, precision, and recall

Our suggested models surpass, even if subtly, DeepEI in accuracy, precision, and recall both separately for ECFP6 (Figure S1) and MACCS keys (Figure S2) and combined (Figure 2 and Table 2). Accuracy values were the highest of the four classification metrics because accuracy is most aligned with the BCE loss function that was used for training. Mathematical rule was used for rounding predicted fingerprint values from float to integer/boolean.

The metrics for the Lite model were reproducible among

Table 2. Classification metrics for the predictive models.

Metric		DeepEI	Full	Lite
Mean	Accuracy	0.89	0.92	0.91
	Precision	0.77	0.87	0.86
	Recall	0.70	0.73	0.71
	F1	0.73	0.79	0.77
Median	Accuracy	0.88	0.91	0.91
	Precision	0.79	0.88	0.87
	Recall	0.73	0.77	0.75
	F1	0.75	0.82	0.80

Table 1. Comparison of inference and training performance (times) of the models.

Model	CPU Inference, s (≈ 26,000 molecules)	GPU Training, s (≈ 200,000 molecules)	Average speed increase vs DeepEI
DeepEI	480	36 000 / 8 epochs	1x
Full	13	1200 / 100 epochs	34x
Lite	3	360 / 100 epochs	130x

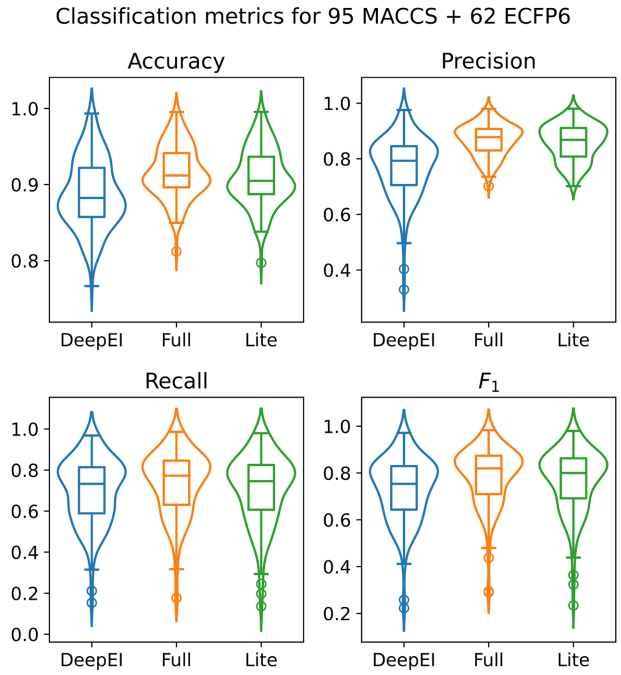

Figure 2. Classification metrics for predictive models. The boxes show the quartiles of the distribution (the line inside is median), the whiskers extend to points that lie within the 1.5 interquartile ranges of the lower and upper quartile. The violin plots show distribution of values approximated by kernel density estimation.

the repeated runs with 6 different seeds used for obtaining training, validation and test datasets (see Figure S3 and Table S1).

Example of usage

MFs are most useful when an unknown compound is absent in available databases. The first step of the MF-based identification approach is obtaining a list of SMILES for potential candidates which can be taken from general purpose chemical databases (e.g., PubChem or ChemSpider) and additionally expanded algorithmically (e.g. by replacing active hydrogens with trimethylsilyl groups). Then MFs can be calculated for all potential candidates using available tools (e.g., RdKit) to form a fully generated database of MFs. MFs for the unknown compound are predicted from its mass spectrum using the Lite version of the model. Then these MFs are searched against the generated database using Tanimoto similarity (or Jaccard index) to give the researcher more information about the unknown compound.

Experimental EI mass spectra of 3,3-diethoxypropiophenone and bis(2,4-dinitrophenyl)ether (Table 3) from MassBank (that are not present in the NIST mainlib) were selected to illustrate an MF-based identification approach. These mass spectra were searched against the NIST mainlib

Table 3. Results of searching for compounds from MassBank not present in NIST mainlib using mass spectra (Similarity search in MS Search) and MFs (Tanimoto similarity).

Original Compound	Top Compound by Similarity Search	Top Compound by MFs search
3,3-diethoxypropiophenone	β-Benzil monoxime	3,3-dimethoxy propiophenone
bis(2,4-dinitro phenyl)ether	1,2-dimethoxy-4-nitroso benzene	2,4-dinitro-1-(O-nitrophenoxy)benzene

using the Similarity algorithm (recommended by its developers for cases when the unknown compound is absent from the database). In both cases the top candidates significantly differed from the original structures. An alternative approach involved predicting MFs by the Lite model, followed by searching against the MFs database calculated by RdKit for all compounds from NIST mainlib. The top candidates returned using this approach had structures quite similar to the correct ones.

Conclusions

In this work we suggested two models (Lite and Full) with fully-connected architecture that were capable of predicting several types of molecular fingerprints (ECFP6 and MACCS keys) from EI mass spectra using a single multi-output neural network. The lightweight and performant Lite version has better accuracy (0.91 vs 0.89), precision (0.86 vs 0.77), and recall (0.71 vs 0.70) than the DeepEI approach. It is also more than 100 times faster in training and inference. Moreover, it is more easily managed than 157 separate DeepEI models. The Full architecture can be used as a base to develop fine-tuned solutions for predicting other types of fingerprints, much like it can be transformed into the respective Lite version using the approach suggested in the article. More than that, the ideas behind the architectures: choosing the number of layers, use of Dropout and BatchNorm layers were discussed, different activation functions (ReLU, Leaky ReLU, SiLU) were tested and the possibility of using convolutional layers was investigated. It was demonstrated that the Lite version of the neural network was stable during repeated runs with different seeds for train-validation-test splits. Taking into account its accuracy, precision, recall, inference and training speed, we believe that the Lite version can be used as a part of a GC/

MS data processing suite for prediction of molecular fingerprints from the EI mass spectra. These molecular fingerprints predicted by the Lite model from experimental EI mass spectra of unknown compounds can be compared against MFs calculated for known molecules and used in ranking algorithms to shorten the list of possible candidates and find molecules with similar structures.

Acknowledgments

This work was supported by a grant from the Russian Science Foundation (Grant No. 22-13-00266) for the Frumkin Institute of Physical Chemistry and Electrochemistry, Russian Academy of Sciences.

Notes

†Electronic Supplementary Information (ESI) available: See https://github.com/mkhrisanfov/EI2FP

References

1. A. Y. Sholokhova, D. D. Matyushin, O. I. Grinevich, S. A. Borovikova, A. K. Buryak, *Molecules* **2023**, *28*, 3409. https://doi.org/10.3390/molecules28083409.
2. C. Ruttkies, S. Neumann, S. Posch, *BMC Bioinformatics* **2019**, *20*, 376. https://doi.org/10.1186/s12859-019-2954-7.
3. .F. Wang, J. Liigand, S. Tian, D. Arndt, R. Greiner, D. S. Wishart, *Anal. Chem.* **2021**, *93*, 11692. https://doi.org/10.1021/acs.analchem.1c01465.
4. J. N. Wei, D. Belanger, R. P. Adams, D. Sculley, *ACS Cent. Sci.* **2019**, *5*, 700. https://doi.org/10.1021/acscentsci.9b00085.
5. G. Landrum, P. Tosco, B. Kelley, Ric, sriniker, D. Cosgrove, gedeck, R. Vianello, NadineSchneider, E. Kawashima, D. N, G. Jones, A. Dalke, B. Cole, M. Swain, S. Turk, AlexanderSavelyev, A. Vaucher, M. Wójcikowski, I. Take, D. Probst, K. Ujihara, V. F. Scalfani, guillaume godin, A. Pahl, F. Berenger, JLVarjo, R. Walker, jasondbiggs, strets123, rdkit/rdkit: 2023_03_1 (Q1 2023) Release, Zenodo, **2023**. https://doi.org/10.5281/zenodo.7880616.
6. E. L. Willighagen, J. W. Mayfield, J. Alvarsson, A. Berg, L. Carlsson, N. Jeliazkova, S. Kuhn, T. Pluskal, M. Rojas-Chertó, O. Spjuth, G. Torrance, C. T. Evelo, R. Guha, C. Steinbeck, *J. Cheminformatics* **2017**, *9*, 33. https://doi.org/10.1186/s13321-017-0220-4.
7. H. Moriwaki, Y.-S. Tian, N. Kawashita, T. Takagi, *J. Cheminformatics* **2018**, *10*, 4. https://doi.org/10.1186/s13321-018-0258-y.
8. C. W. Yap, *J. Comput. Chem.* **2011**, *32*, 1466. https://doi.org/10.1002/jcc.21707.
9. Dragon 7.0 - Kode Chemoinformatics, **2019**.
10. A. Mauri, in *Ecotoxicological QSARs*, ed. by Kunal Roy, Springer US, New York, NY, **2020**, pp. 801–820. https://doi.org/10.1007/978-1-0716-0150-1_32.
11. D. D. Matyushin, A. Y. Sholokhova, A. K. Buryak, *Int. J. Mol. Sci.* **2021**, *22*, 9194. https://doi.org/10.3390/ijms22179194.
12. R. L. Zhu, E. Jonas, *Anal. Chem.* **2023**, *95*, 2653. https://doi.org/10.1021/acs.analchem.2c02093.
13. E. S. Fedorova, D. D. Matyushin, I. V. Plyushchenko, A. N. Stavrianidi, A. K. Buryak, *J. Chromatogr. A* **2022**, *1664*, 462792. https://doi.org/10.1016/j.chroma.2021.462792.
14. PubChem, Data Specification, https://pubchem.ncbi.nlm.nih.gov/docs/data-specification.
15. J. Klekota, F. P. Roth, *Bioinformatics* **2008**, *24*, 2518. https://doi.org/10.1093/bioinformatics/btn479.
16. J. L. Durant, B. A. Leland, D. R. Henry, J. G. Nourse, *J. Chem. Inf. Comput. Sci.* **2002**, *42*, 1273. https://doi.org/10.1021/ci010132r.
17. D. Rogers, M. Hahn, *J. Chem. Inf. Model.* **2010**, *50*, 742. https://doi.org/10.1021/ci100050t.
18. H. L. Morgan, *J. Chem. Doc.* **1965**, *5*, 107. https://doi.org/10.1021/c160017a018.
19. Daylight Theory: Fingerprints, https://www.daylight.com/dayhtml/doc/theory/theory.finger.html.
20. K. Dührkop, H. Shen, M. Meusel, J. Rousu, S. Böcker, *Proc. Natl. Acad. Sci.* **2015**, *112*, 12580. https://doi.org/10.1073/pnas.1509788112.
21. F. Qiu, Z. Lei, L. W. Sumner, *Anal. Chim. Acta* **2018**, *1037*, 316. https://doi.org/10.1016/j.aca.2018.03.052.
22. H. Ji, H. Deng, H. Lu, Z. Zhang, *Anal. Chem.* **2020**, *92*, 8649. https://doi.org/10.1021/acs.analchem.0c01450.
23. X. Fan, Z. Xu, H. Zhang, D. Liu, Q. Yang, Q. Tao, M. Wen, X. Kang, Z. Zhang, H. Lu, *Talanta* **2022**, *244*, 123415. https://doi.org/10.1016/j.talanta.2022.123415.
24. N. F. de Jonge, K. Mildau, D. Meijer, J. J. R. Louwen, C. Bueschl, F. Huber, J. J. J. van der Hooft, *Metabolomics* **2022**, *18*, 103. https://doi.org/10.1007/s11306-022-01963-y.
25. X. Xue, H. Sun, M. Yang, X. Liu, H.-Y. Hu, Y. Deng, X. Wang, *Anal. Chem.* **2023**, *95*, 13733. https://doi.org/10.1021/acs.analchem.3c02540.
26. M. Seifrid, R. Pollice, A. Aguilar-Granda, Z. Morgan Chan, K. Hotta, C. T. Ser, J. Vestfrid, T. C. Wu, A. Aspuru-Guzik, *Acc. Chem. Res.* **2022**, *55*, 2454. https://doi.org/10.1021/acs.accounts.2c00220.
27. S. F. Baygi, D. K. Barupal, *J. Cheminformatics* **2024**, *16*, 8. https://doi.org/10.1186/s13321-024-00804-5.
28. S. Galati, M. Di Stefano, E. Martinelli, M. Macchia, A. Martinelli, G. Poli, T. Tuccinardi, *Int. J. Mol. Sci.* **2022**, *23*, 2105. https://doi.org/10.3390/ijms23042105.
29. U. W. Liebal, A. N. T. Phan, M. Sudhakar, K. Raman, L. M. Blank, *Metabolites* **2020**, *10*, 243. https://doi.org/10.3390/metabo10060243.
30. G. Hu, M. Qiu, *Nat. Prod. Rep.* **2023**. https://doi.org/10.1039/D3NP00025G.
31. F. Herrera-Rocha, M. Fernández-Niño, J. Duitama, M. P. Cala, M. J. Chica, L. A. Wessjohann, M. D. Davari, A. F. G. Barrios, *FlavorMiner: A Machine Learning Platform for Extracting Molecular Flavor Profiles from Structural Data*, ChemRxiv, **2024**. https://doi.org/10.26434/chemrxiv-2024-821xm.
32. M. A. Stravs, K. Dührkop, S. Böcker, N. Zamboni, *Nat. Methods* **2022**, *19*, 865. https://doi.org/10.1038/s41592-

022-01486-3.
33. D. Elser, F. Huber, E. Gaquerel, *Mass2SMILES: deep learning based fast prediction of structures and functional groups directly from high-resolution MS/MS spectra*, bioRxiv, **2023**. https://doi.org/10.1101/2023.07.06.547963.
34. S. F. Baygi, D. K. Barupal, *J. Cheminformatics* **2024**, *16*, 8. https://doi.org/10.1186/s13321-024-00804-5.
35. S. Gao, H. Y. K. Chau, K. Wang, H. Ao, R. S. Varghese, H. W. Ressom, *Metabolites* **2022**, *12*, 605. https://doi.org/10.3390/metabo12070605.
36. A. Paszke, S. Gross, F. Massa, A. Lerer, J. Bradbury, G. Chanan, T. Killeen, Z. Lin, N. Gimelshein, L. Antiga, A. Desmaison, A. Kopf, E. Yang, Z. DeVito, M. Raison, A. Tejani, S. Chilamkurthy, B. Steiner, L. Fang, J. Bai, S. Chintala, Advances in Neural Information Processing Systems.
37. C. R. Harris, K. J. Millman, S. J. van der Walt, R. Gommers, P. Virtanen, D. Cournapeau, E. Wieser, J. Taylor, S. Berg, N. J. Smith, R. Kern, M. Picus, S. Hoyer, M. H. van Kerkwijk, M. Brett, A. Haldane, J. F. del Río, M. Wiebe, P. Peterson, P. Gérard-Marchant, K. Sheppard, T. Reddy, W. Weckesser, H. Abbasi, C. Gohlke, T. E. Oliphant, *Nature* **2020**, *585*, 357. https://doi.org/10.1038/s41586-020-2649-2.
38. The pandas development team, pandas-dev/pandas: Pandas, Zenodo, **2024**. https://doi.org/10.5281/zenodo.10957263.
39. J. D. Hunter, *Comput. Sci. Eng.* **2007**, *9*, 90. https://doi.org/10.1109/MCSE.2007.55.
40. M. L. Waskom, *J. Open Source Softw.* **2021**, *6*, 3021. https://doi.org/10.21105/joss.03021.
41. mkhrisanfov/EI2FP: Efficient prediction of molecular fingerprints from electron ionization mass-spectra, https://github.com/mkhrisanfov/EI2FP.
42. V. Nair, G. E. Hinton, Proceedings of the 27th international conference on machine learning (ICML-10), 807–814.
43. A. L. Maas, A. Y. Hannun, A. Y. Ng, others, ICML'13: Proceedings of the 30th International Conference on International Conference on Machine Learning, 3.
44. D. Hendrycks, K. Gimpel, *Gaussian Error Linear Units (GELUs)*, arXiv:1606.08415, **2023**. https://doi.org/10.48550/arXiv.1606.08415.
45. P. Ramachandran, B. Zoph, Q. V. Le, *Searching for Activation Functions*, arXiv:1710.05941, **2017**. https://doi.org/10.48550/arXiv.1710.05941.
46. S. Elfwing, E. Uchibe, K. Doya, *Sigmoid-Weighted Linear Units for Neural Network Function Approximation in Reinforcement Learning*, arXiv:1702.03118, **2017**. https://doi.org/10.48550/arXiv.1702.03118.
47. D. D. Matyushin, A. Yu. Sholokhova, A. K. Buryak, *J. Chromatogr. A* **2019**, *1607*, 460395. https://doi.org/10.1016/j.chroma.2019.460395.

