analytical chemistry

Letter

Cite This: Anal. Chem. XXXX, XXX, XXX−XXX

pubs.acs.org/ac

# Deep Learning for the Precise Peak Detection in High-Resolution LC−MS Data

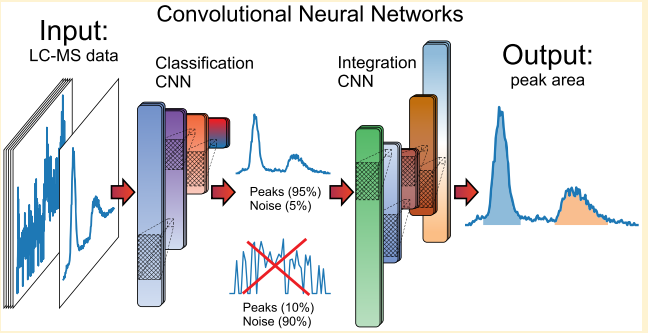
Arsenty D. Melnikov,[†,‡] Yuri P. Tsentalovich,[†,‡] and Vadim V. Yanshole*[,†,‡]

[†]International Tomography Center SB RAS, Institutskaya 3a, Novosibirsk 630090, Russia

[‡]Novosibirsk State University, Pirogova 2, Novosibirsk 630090, Russia

Supporting Information

**ABSTRACT:** This letter is devoted to the application of machine learning, namely, convolutional neural networks to solve problems in the initial steps of the common pipeline for data analysis in metabolomics. These steps are the peak detection and the peak integration in raw liquid chromatography−mass spectrometry (LC−MS) data. Widely used algorithms suffer from rather poor precision for these tasks, yielding many false positive signals. In the present work, we developed an algorithm named *peakonly*, which has high flexibility for the detection or exclusion of low-intensity noisy peaks, and shows excellent quality in the detection of true positive peaks, approaching the highest possible precision. The current approach was developed for the analysis of high-resolution LC−MS data for the purposes of metabolomics, but potentially it can be applied with several adaptations in other fields, which utilize high-resolution GC− or LC−MS techniques. *Peakonly* is freely available on GitHub (https://github.com/arseha/peakonly) under an MIT license.






Liquid chromatography in conjunction with mass spectrometry (LC−MS) is one of the most sought after analytical platforms in metabolomics. Despite the wide variety of LC−MS based metabolomics applications[1] and recent analytical hardware developments, the treatment of LC−MS data still encounters some problems. One of the most critical bottlenecks is the raw data processing. LC−MS raw data usually consists of thousands of raw MS spectra placed one after another; each spectrum has its own sequence number and this number increases with the retention time (RT). These data typically include thousands of signals (features), which make the manual data processing almost impossible. Current pipelines for the automatic LC−MS data processing usually consist of the following steps: (1) the detection of regions of interest (ROIs), (2) the detection of chromatographic peaks followed by their integration, (3) peak matching (grouping) for all samples in the batch, and (4) clustering of peaks belonging to the same metabolite into one group by the annotation of corresponding adduct and fragment ions.

*XCMS* and *MZmine 2* are the most widely used open source software for LC−MS raw data processing[2,3] which are able to perform all four steps and give to a user a fairly complete table of peaks found in spectra and their integral intensities (i.e., peak area) for each sample. Unfortunately, these applications usually tend to produce many false positive signals, which can lead to false scientific discoveries or complicate the interpretation of real ones.[4]

The peak detection and integration problems can be solved by the methods of machine learning, such as artificial neural networks (NNs). Recent advances of deep learning (DL), particularly of convolutional neural networks (CNNs) in computer vision tasks[5−7] such as image classification and object detection, and rapid development of DL frameworks, e.g., *PyTorch*,[8,9] make it possible to adapt DL methods for solving problems in bioinformatics. In the recent paper,[10] authors applied NN for predicting whether the specific coordinate is the center of a peak. This approach can hardly be considered as the solution of the problem since it is rather time-consuming (about an hour for each sample), predicts only the coordinate of the center of a peak, and does not calculate the peak area. The latter can be an even harder task, especially in the case of overlapping peaks. *DeepIso* is another interesting example of DL application for LC−MS data treatment.[11] Authors applied CNN to detect peptide features and estimate their abundances. Another paper[12] describes the use of machine learning (but not DL) for the optimization of the peak detection in GC−MS metabolomics data yielding high-quality features. The very recent paper by Kantz et al.[13] is devoted to the application of DL for excellent noise filtering in LC−MS metabolomics data. Two latter algorithms[12,13] treat the initially preprocessed data, the peaks already detected and integrated by existing software (*XCMS*, *MZmine 2*, and several others). However, these approaches do not solve problems of incorrectly integrated or missed peaks.

**Received:** October 22, 2019
**Accepted:** December 16, 2019
**Published:** December 16, 2019




DOI: 10.1021/acs.analchem.9b04811
Anal. Chem. XXXX, XXX, XXX−XXX

The present work is devoted to the application of CNNs to solve problems in the initial steps of the common processing pipeline, peak detection and integration in untreated (raw) LC−MS data. As the result, we developed an algorithm *peakonly*, which implements only the first two steps of the pipeline. For ROI detection, we adopted and slightly modified the *centWave* ROI detection algorithm.[14] Then we built a CNN, which classifies ROI into three classes: (1) ROI does not contain peaks, only noise; (2) ROI contains one or more peaks; (3) ROI contains something like a peak, but special attention from a specialist is required. Peak integration we considered as a segmentation problem, whether the point in a ROI corresponds to the peak. To solve this problem, we constructed a second CNN with a similar to *U-Net* basic idea.[7] The source code for the algorithm and the automation script, which one can use for the treatment of a single MS1 centroid high-resolution LC−MS file in mzML format is deposited on GitHub (detailed instruction is available at the project repository).[15] *Peakonly* was shown to work with liquid chromatography−quadrupole-time-of-flight (LC−Q-TOF) data from our lab and with several examples of high-resolution LC−MS data (LC−Q-TOF, LC−Orbitrap, liquid chromatography−ion cyclotron resonance (LC−ICR)) from the MetaboLights repository. Therefore, the algorithm is suitable for the application to the wide variety of LC−MS data; nevertheless, significantly different data (e.g., GC−MS) may demand additional training of the CNNs to improve the output data quality.

## DATA MINING AND CNN DESCRIPTION

The code is written in Python v.3.5. Neural networks were constructed and trained using *PyTorch* v.1.2.[9] Additional libraries used in this work are listed in the "requirements.txt" file in the project repository on GitHub.[15] The detailed description of CNNs including their architectures and training process is provided in the Supporting Information.

**Data Mining.** The raw data from the LC−MS instrument were converted into centroid spectra of mzML file format. A ROI detection algorithm similar to *centWave*[14] was implemented in Python using *pymzML* for mzML data reading.[16] In the *centWave*, the appearance of zero points led to the immediate ROI termination. The only modification in our algorithm compared to *centWave* is that we added the possibility that ROI can include zero points but no more than three in a row. In fact, the number of zero points in a row is a variable parameter, and it can be adapted for different types of data. The purpose of this modification is the following: noise sometimes can randomly form something looking like a peak; without vicinity points, it is hard to classify whether the region contains peak or noise, even for a human expert.

To build a data set, we manually annotated more than 4000 ROIs. To get examples of ROIs, we used in-house LC−Q-TOF MS spectra: the spectra of extracts from the human serum and from the lens of the fresh-water fish *S. lucioperca*, obtained in the positive mode using two types of liquid chromatography−reversed-phase high-performance and hydrophilic interaction chromatography with either a 1 or 3 Hz MS scan rate. The detailed description of the samples and instrumental setup is given in papers[17,18] and in the Supporting Information. The diversity of the data was required to expand the variety of peak shapes and thereby enhance the generalizing ability of the neural network.

Every ROI was attributed to one of the previously described classes: ROI does not contain a peak, only noise (class 1); ROI contains one or more peaks (class 2); ROI contains something like a peak, but special attention from a specialist is required (class 3). The most difficult problem was the separation between noisy peaks of small intensity (class 2) and signals too noisy or too small or strangely looking to be attributed to a peak (class 3). Thus, the border between classes is rather blurry, and even very similar peaks in the resulting data set could be assigned to different classes (Figure 1). Later we will show that this leads to the assignment of low-intensity noisy peaks to the second and third classes with almost equal probabilities. That provides an additional flexibility for users to include or exclude low-intensity peaks by changing the threshold for the classification probability.

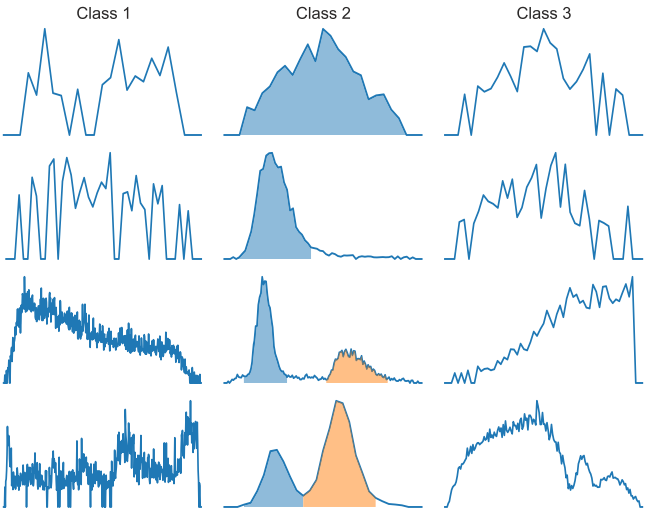



**Figure 1.** Examples of ROIs from each class. Class 1, ROIs classified as noise; Class 2, ROIs classified as one or more peaks; Class 3, uncertain peaks. Blue and orange fillings represent the regions for the peak integration. ROIs were taken from the training set.

For each ROI assigned to the second class, we additionally manually annotated the number of peaks within the ROI. For each peak, we have manually marked the starting and ending points, and the intersection points between peaks within ROI. Subsequently this information was used to train the CNN for the correct integration of peaks.

After the ROI annotation, we have randomly split our data into three sets: training, validation, and test sets. The validation and test sets consisted of 200 examples from each class, while the training set included all other ROIs (∼900 of class 1, ∼850 of class 2, ∼1350 of class 3).

We also used synthetic data for training. The synthetic data included random noise, one or more asymmetrical peaks (two halves of a gauss with different sigma values) with the noise of different amplitude, and even peaks similar to signals from class 3. Each training batch included approximately 70% of synthetic data. The Python code for the synthetic data generator is also available on GitHub.[15] The examples of the generated ROIs can be found in the Supporting Information (Figure S4).

**CNN for ROI Classification.** We took the advantage of the intrinsic ability of CNN to classify different objects (text, images, audio, video, etc.) and applied CNN for the classification of ROIs into three previously described classes. Before the input to the neural network, all ROIs were


DOI: 10.1021/acs.analchem.9b04811
Anal. Chem. XXXX, XXX, XXX−XXX

subjected to the data preprocessing to make uniform data input for CNN. The length of each ROI was linearly interpolated, transforming the ROI size to 256 points. The signal intensities in ROIs were scaled to unity at the maximum. Thus, our network knows nothing about the intensities and makes predictions based on the peak shape only. The output of the CNN for ROI classification is the calculated probabilities (from 0 to 1) of the ROI assignment to each of three classes; the sum of three probability values is equal to 1 (Figure S1). The accuracy of the final model was estimated with the use of the test set, and it achieved approximately 87%. For more detailed analysis, the confusion matrix was built (Figure S2). The matrix shows that our model rarely confuses peaks with noise, only 0.5% of manually labeled peaks are classified as noise. The majority of errors of the model correspond to the incorrect assignment of ROIs to class 3. Further analysis showed that in most error cases, the ROIs were quite unobvious for the assignment even for a human expert. Usually, in such cases the model predicted fairly similar probabilities of the ROI assignment to different classes.

**CNN for Peak Integration.** We considered the determination of the LC peak region as a segmentation problem. Typically, segmentation results in the attribution of a part of an image or a signal to a certain object. For the better determination of the peak borders, we predicted not only the peak regions, but also the separation regions. The basic structural idea of the second CNN for the peak integration is similar to *U-Net*,[7] which propagates feature maps from the contraction part into the expansion part. The *U-net* is usually used for the fast and precise segmentation of images. However, we made some architectural modifications and dramatically reduced the number of parameters since ROIs are much simpler than images (Figure S3). The final model achieved about 0.88 in the intersection over union (IoU)[19] metric for the prediction of the separation regions and 0.85 for the prediction of the peak regions in the test set.

In the current work, NVIDIA RTX 2060 6GB was used to train the neural networks. Nevertheless, since the size of the input data is small (each ROI is converted to 256 points) and the sizes of the networks are optimized for fast data processing, one may use standard CPU with approximately 6 GB RAM to repeat the whole training process. CPU specification affects only the training time. Training with the use of a fairly modern CPU (~6 cores, ~2500 MHz) takes less than an hour in total.

## EVALUATION OF THE ALGORITHM

The quality of peak area calculation was evaluated with the use of two approaches. First, we compared the manually annotated areas and the areas found by *peakonly* using the data from the test set. The average relative error of areas ($\Delta S/S$) was about 4%. Second, we used data from MetaboLights repository (MTBLS234) and corresponding paper[20] where authors have spiked different concentrations of reserpine and terfenadine into the human plasma samples. A good agreement was found between the published linear dependencies (area vs concentration) and the peak areas re-evaluated by our algorithm (Figures S13–S16).

To evaluate the final quality of neural networks for the peak detection in raw LC–MS data, the pipeline with the following three steps was applied: (1) ROI detection, (2) ROI classification, (3) peak separation and integration. For the second and third steps of the pipeline, two neural networks described above were used. The method developed in this work was compared with the performance of the widely used in the metabolomics society software, *XCMS Online* (version 2.7.2 with *XCMS* version 1.47.3), and *MZmine 2* (version 2.52). For the comparison of these methods, the LC–MS data obtained from the equimolar mixture of 27 chemical standards of metabolites were used;[18] the percentage of the true positive peaks among detected features (precision) as well as the fraction of missed peaks (1 − recall) in the data were evaluated. The results of the comparison of *peakonly* and *XCMS Online* are presented in the text, while similar results obtained for *peakonly* vs *MZmine 2* comparison are given in the Supporting Information (Figures S10–S12).

For the peak detection with the use of neural networks, the following parameters were used: $m/z$ window, ±0.005 Da; minimal length of ROI, 12 points; minimal peak length, 6 points; maximal number of zero points in a row, 3. All these parameters affect only the ROI detection and postprocessing and do not influence the performance of neural networks. We did not consider peaks with the integration region (peak width) of less than 6 points. By our opinion, the LC peak should consist of at least six to ten data points to have a defined shape and to enable confident quantification based on the integration of its area. However, one can adjust this parameter at the input to *peakonly*. As a post processing, we also removed peaks with the RT length of more than 3 min (equivalent to 180 points for 1 Hz rate of the data acquisition). The entire processing of a single LC–MS data file took less than 2 min.

The following parameters for *XCMS Online* were selected: *centWave* method, *ppm*, 5; *snthr*, 6; *mzdiff*, 0.005; *minimum peak width*, 15 points; *maximum peak width*, 120 points; *prefilter peaks*, 3; *prefilter intensity*, 300; *noise*, 100.

It is important to note that although the chemical mixture used in this work contained only 27 metabolites, every compound gave much more than one signal due to the isotopic distribution, ion fragmentation, and adduct formation. The impurities contained in the LC–MS system and in the solution also gave a substantial contribution into the total number of detected features. The neural network approach found 926 features, while *XCMS Online* found 1764 features (Figure 2). We assumed that both methods found the same peak if the median RT calculated by *XCMS Online* was within the integration boundaries predicted by *peakonly* and the median $m/z$ value (*XCMS Online*) was within the range of $m/z$ values in corresponding ROI (*peakonly*). Using this algorithm, we found that 838 peaks are detected by both methods. Thereby, our approach detected 88 unique features. Subsequent analysis revealed that 14 of them were actually the false positive signals (noise, Figure S9), while the rest of the 74 were real peaks. We have inspected all 838 features found by both methods; 17 of them turned out to be noise. Therefore, our approach achieves the precision of $(926 - 14 - 17)/926 = 97\%$. That is indeed a high value since the precision of the existing algorithms without additional noise filtering[12,13] usually ranges from 0.5 to 0.8.[21]

We also analyzed the intersection of features found by *XCMS Online* and the features, which were assigned to class 3 (uncertain peaks) by *peakonly*. We did not determine the integration boundaries for these features and only considered the ROIs no longer than 1 min (equivalent to 60 points). In total, our algorithm found 444 such features, and 305 of them intersected with features found by *XCMS Online*. Thereby, *XCMS Online* detected 621 unique features, which were not


DOI: 10.1021/acs.analchem.9b04811
*Anal. Chem.* XXXX, XXX, XXX–XXX

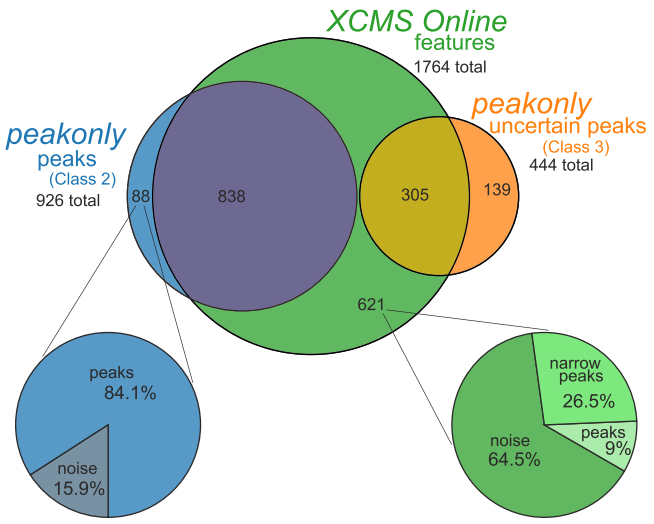



**Figure 2.** Venn diagram showing the number of features found by *XCMS Online* and *peakonly* in a mixture of known metabolites. Features classified as peaks (class 2) and uncertain peaks (class 3) are shown separately. Percentage of noise, narrow peaks, and peaks found only by *XCMS Online* was calculated from a random sampling of 200 out of 621 features.

detected by our algorithm and were not classified as uncertain peaks (Figure 2). We randomly selected 200 features out of these 621 ones and manually classified them into 3 groups: (1) noise, including peaks with incorrect integration boundaries (50 or more percent of the area was not calculated), (2) peaks, (3) very narrow peaks, with the integration width of fewer than 6 points or ROI consisting of fewer than 12 points. The examples of unique features found by *XCMS Online* are provided in the Supporting Information (Figures S5–S7). A total of 129 (64%) of these features were classified as a noise, 18 as peaks (9%), and 53 as narrow peaks (26.5%). Since we deliberately did not take into account narrow peaks and uncertain peaks with noisy shape in order to detect confidently only true positive peaks, the developed approach turned out to miss about 60 peaks: 9% from 621 unique features detected by *XCMS Online* and missed by *peakonly*. The majority of these peaks are small-intensity noisy signals (Figures S7, S9, and S12), and the detection of such peaks by *peakonly* could easily be improved if needed; however, this may increase the amount of false positive peaks. Thereby, the recall of the method could be estimated as $(926 - 14 - 17)/(926 + 80) = 89\%$. The analysis of true positive peaks, found by *XCMS Online* and not found by *peakonly* is given in the Supporting Information (Figure S8). The summary of the method comparison is shown in Figure 2.

The algorithm was tested to process not only in-house LC–Q-TOF data but also external LC–Q-TOF (MTBLS234), LC–Orbitrap (MTBLS404), and LC–ICR (MTBLS210) data; the results of tests are provided in the Supporting Information (Figures S13–S18). *Peakonly* also processes high-resolution GC–MS data (MTBLS27), but these data differ from the LC–MS data and therefore the output quality is not very high. For such specific data, additional training of CNNs is required to improve output quality.

## CONCLUSIONS

We developed the new feature detection method based on the use of convolutional neural networks. The instructions for the processing of single LC–MS file are provided in the Supporting Information and in the project repository on GitHub. The developed algorithm consists of three main steps: (1) ROI detection, (2) ROI classification, (3) peak detection and integration. For the second and third steps, we constructed and trained two neural networks. The method demonstrated the ability to detect true positive peaks with high precision and to substantially reduce the amount of noise. One of the advantages of the developed method is the ability to recognize the peaks with a noisy shape. Thus, one gets a table of fairly good peaks at the very first stage of the raw data processing (peak detection). This can greatly simplify the work of an analyst. The detailed analysis revealed that the developed algorithm approaches the highest possible precision (97%) in the detection of true positive peaks. Unfortunately, there are also peaks, which the current approach misses. The endeavor to detect all peaks in raw data is always a balance between the detection of low-intensity peaks and removing noise. Even a skilled human expert not always can draw a clear boundary between these two groups. In the near future, we plan to work on improving the quality of the developed method. One of the advantages of using neural networks is its high flexibility. One may significantly improve the quality of models by adding new data for training or even modifying the current architecture. If needed, the quality of the detected peaks could also be improved by the increasing of the probability threshold during the ROI classification. *Peakonly* has been shown to detect and integrate high-quality peaks in single LC–MS files. The embedding of subsequent analysis steps, peak matching, zero filling, and clustering between samples in the sample batch is under active development in our lab.

The developed algorithm takes less than 2 min for a typical mzML file (~50 Mb) processing, which is quite acceptable for processing of a single sample. Nevertheless, some algorithms implemented in *peakonly* can be significantly accelerated by code refactoring, and the single LC–MS file processing is expected to last for less than 30 s. We believe that the current work will attract attention to the application of neural networks to solve problem of the peak detection in raw data. We also believe that the cooperation with other laboratories can significantly improve the method quality by expanding the training data using the examples from other LC–MS devices. The current approach was developed for the processing of high-resolution LC–MS data for the purposes of metabolomics, but it can be applied with several adaptations in other fields, which utilize high-resolution GC– or LC–MS techniques.

## ASSOCIATED CONTENT

### Supporting Information

The Supporting Information is available free of charge at https://pubs.acs.org/doi/10.1021/acs.analchem.9b04811.

Description of samples and instrumental setup, description of CNN architectures and training process, pipeline for data analysis, and evaluation of the algorithm (PDF)

## AUTHOR INFORMATION

### Corresponding Author

*E-mail: vadim.yanshole@tomo.nsc.ru.

### ORCID

Arsenty D. Melnikov: 0000-0003-1167-9103



DOI: 10.1021/acs.analchem.9b04811
*Anal. Chem.* XXXX, XXX, XXX–XXX

Yuri P. Tsentalovich: 0000-0002-1380-3000
Vadim V. Yanshole: 0000-0003-1512-3049

### Notes

The authors declare no competing financial interest.
*Peakonly* is freely available on GitHub (https://github.com/arseha/peakonly) under an MIT license.

## ■ ACKNOWLEDGMENTS

The algorithm development was funded by the Russian Foundation for Basic Research (RFBR), Project 18-29-13023. The collection of LC−MS data was supported by RFBR (Project 19-04-00092) and RFBR and the government of the Novosibirsk region (Project 18-415-543006). We thank the Ministry of Science and Higher Education of the RF for the access to MS equipment.

## ■ REFERENCES

(1) Wehrens, R.; Salek, R. *Metabolomics: Practical Guide to Design and Analysis*; CRC Press, 2019.
(2) Gowda, H.; Ivanisevic, J.; Johnson, C. H.; Kurczy, M. E.; Benton, H. P.; Rinehart, D.; Nguyen, T.; Ray, J.; Kuehl, J.; Arevalo, B.; et al. *Anal. Chem.* **2014**, *86* (14), 6931−6939.
(3) Pluskal, T.; Castillo, S.; Villar-Briones, A.; Orešič, M. *BMC Bioinf.* **2010**, *11*, 395.
(4) Myers, O. D.; Sumner, S. J.; Li, S.; Barnes, S.; Du, X. *Anal. Chem.* **2017**, *89* (17), 8689−8695.
(5) Ren, S.; He, K.; Girshick, R.; Sun, J. Faster R-Cnn: Towards Real-Time Object Detection with Region Proposal Networks. In *Advances in Neural Information Processing Systems*; 2015; pp 91−99.
(6) Krizhevsky, A.; Sutskever, I.; Hinton, G. E. Imagenet Classification with Deep Convolutional Neural Networks. In *Advances in Neural Information Processing Systems*; 2012; pp 1097−1105.
(7) Ronneberger, O.; Fischer, P.; Brox, T. U-Net: Convolutional Networks for Biomedical Image Segmentation. In *MICCAI 2015: Medical Image Computing and Computer-Assisted Intervention − MICCAI 2015. Lecture Notes in Computer Science*; Navab, N., Hornegger, J., Wells, W., Frangi, A., Eds.; Springer: Cham, Switzerland, 2015; Vol. *9351*, pp 234−241. DOI: 10.1007/978-3-319-24574-4_28.
(8) Paszke, A.; Gross, S.; Chintala, S.; Chanan, G.; Yang, E.; DeVito, Z.; Lin, Z.; Desmaison, A.; Antiga, L.; Lerer, A. Automatic Differentiation in PyTorch. In *31st Conference on Neural Information Processing Systems, NIPS 2017 Workshop Autodiff Submission*, Long Beach, CA, December 9, 2017.
(9) *PyTorch Project*, https://pytorch.org (accessed September 19, 2019).
(10) Woldegebriel, M.; Derks, E. *Anal. Chem.* **2017**, *89* (2), 1212−1221.
(11) Zohora, F. T.; Tran, N. H.; Zhang, X.; Xin, L.; Shan, B.; Li, M. DeepIso: A Deep Learning Model for Peptide Feature Detection. *arXiv* **2017**.1801.01539
(12) Borgsmüller, N.; Gloaguen, Y.; Opialla, T.; Blanc, E.; Sicard, E.; Royer, A. L.; Le Bizec, B.; Durand, S.; Migné, C.; Pétéra, M.; et al. *Metabolites* **2019**, *9* (9), 171.
(13) Kantz, E. D.; Tiwari, S.; Watrous, J. D.; Cheng, S.; Jain, M. *Anal. Chem.* **2019**, *91* (19), 12407−12413.
(14) Tautenhahn, R.; Bottcher, C.; Neumann, S. *BMC Bioinf.* **2008**, *9*, 1−16.
(15) peakonly repository https://github.com/arseha/peakonly (accessed Sep 19, 2019).
(16) Kösters, M.; Leufken, J.; Schulze, S.; Sugimoto, K.; Klein, J.; Zahedi, R. P.; Hippler, M.; Leidel, S. A.; Fufezan, C. *Bioinformatics* **2018**, *34* (14), 2513−2514.
(17) Yanshole, V. V.; Yanshole, L. V.; Zelentsova, E. A.; Tsentalovich, Y. P. *Metabolites* **2019**, *9* (5), 95.
(18) Tsentalovich, Y. P.; Yanshole, V. V.; Yanshole, L. V.; Zelentsova, E. A.; Melnikov, A. D.; Sagdeev, R. Z. *Metabolites* **2019**, *9* (11), 264.
(19) Rosebrock, A. *Intersection over Union (IoU) for object detection* https://www.pyimagesearch.com/2016/11/07/intersection-over-union-iou-for-object-detection/ (accessed September 19, 2019).
(20) Kenar, E.; Franken, H.; Forcisi, S.; Wörmann, K.; Häring, H. U.; Lehmann, R.; Schmitt-Kopplin, P.; Zell, A.; Kohlbacher, O. *Mol. Cell. Proteomics* **2014**, *13* (1), 348−359.
(21) Tengstrand, E.; Lindberg, J.; Åberg, K. M. *Anal. Chem.* **2014**, *86* (7), 3435−3442.


DOI: 10.1021/acs.analchem.9b04811
*Anal. Chem.* XXXX, XXX, XXX−XXX