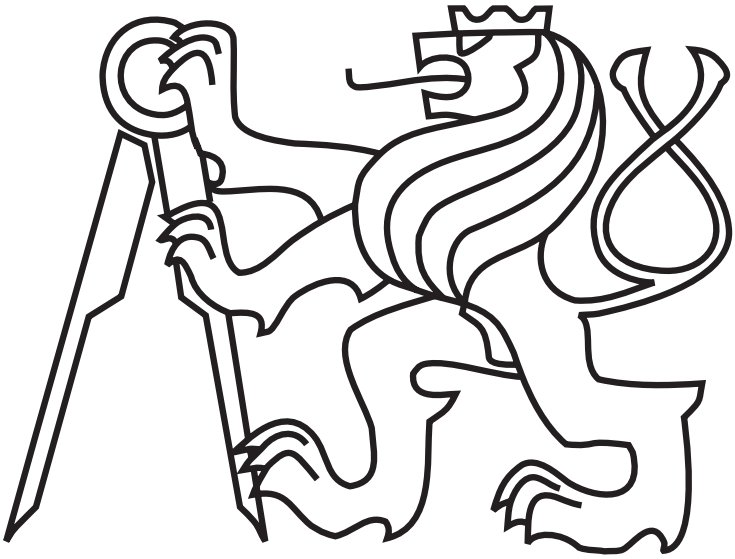
CZECH TECHNICAL UNIVERSITY IN PRAGUE

Faculty of Electrical Engineering

MASTER'S THESIS



Pavel Linder

Out-of-Distribution Detection in Gas Chromatography Mass Spectrometry Data

Department of Cybernetics

Thesis supervisor: **Ing. Bc. Radim Špetlík**

January, 2024

MASTER‘S THESIS ASSIGNMENT

I. Personal and study details

Student's name: **Linder Pavel** Personal ID number: **478055**

Faculty / Institute: **Faculty of Electrical Engineering**

Department / Institute: **Department of Cybernetics**

Study program: **Open Informatics**

Specialisation: **Computer Vision and Image Processing**

II. Master’s thesis details

Master’s thesis title in English:

Out-of-Distribution Detection in Gas Chromatography Mass Spectrometry Data

Master’s thesis title in Czech:

Detekce odchylek mimo distribuci hmotnostních spekter z plynového chromatografu

Guidelines:

1. Familiarize yourself with the published work on the detection of outliers in: (i) image data, and (ii) gas chromatograph mass spectrometer data, if relevant literature is available.
2. Review the literature on algorithms for chemical compound classification.
3. Select and implement or develop a chemical compound classification algorithm, verify its function on data provided by the thesis supervisor.
4. Select and implement or develop at least two algorithms for out-of-distribution detection for use with the classifier from step 2, verify their function on data provided by the thesis supervisor.

Bibliography / sources:

[1] Dan Hendrycks, and Kevin Gimpel. "A Baseline for Detecting Misclassified and Out-of-Distribution Examples in Neural Networks." (2018).
[2] Shiyu Liang, Yixuan Li, and R. Srikant. "Enhancing The Reliability of Out-of-distribution Image Detection in Neural Networks." (2020).
[3] Weitang Liu, Xiaoyun Wang, John D. Owens, and Yixuan Li. "Energy-based Out-of-distribution Detection." (2021).
[4] Dan Hendrycks, Steven Basart, Mantas Mazeika, Mohammadreza Mostajabi, Jacob Steinhardt, and Dawn Song. „Scaling out-of-distribution detection for real-world settings“. (2022).
[5] Tomas Vojir, Jan Sochman, Rahaf Aljundi, and Jiri Matas. "Calibrated Out-of-Distribution Detection with a Generic Representation." (2023).

Name and workplace of master’s thesis supervisor:

Ing. Bc. Radim Špetlík Visual Recognition Group FEE

Name and workplace of second master’s thesis supervisor or consultant:

Date of master’s thesis assignment: **30.08.2023** Deadline for master's thesis submission: **09.01.2024**

Assignment valid until: **16.02.2025**

Ing. Bc. Radim Špetlík	prof. Ing. Tomáš Svoboda, Ph.D.	prof. Mgr. Petr Páta, Ph.D.
Supervisor’s signature	Head of department’s signature	Dean’s signature

III. Assignment receipt

The student acknowledges that the master's thesis is an individual work. The student must produce his thesis without the assistance of others, with the exception of provided consultations. Within the master's thesis, the author must state the names of consultants and include a list of references.

.	
Date of assignment receipt	Student's signature

Declaration

I declare that the presented was developed independently and that I have listed all sources of information used within it in accordance with the methodical instructions for observing the ethical principals in the preparation of university theses.

Prague, data 9. 1. 2024 .....

signature

Acknowledgements

I am profoundly grateful to my supervisor, Ing. Bc. Radim Špetlík, for his valuable feedback, guidance, and advice. His expertise and encouragement have been vital in the completion of this thesis. I also want to extend my appreciation for the access to the computational infrastructure of the 'Research Center for Informatics.' I extend gratitude to my family and my girlfriend, Marina, for their continual support and unwavering belief in my capabilities. Their patience and encouragement have made the challenges more manageable.

Abstract

Chemical compound analysis is an interesting and important problem [1, 2, 3]. Recently, it has been shown [4] that an output of an analytical chemistry tool revealing physical properties of a gas is suitable for gender classification, and identity verification tasks. Correct classification of particular chemical compounds present in a gas sample is crucial for the mentioned tasks. However, the classification heavily relies on proprietary software with high error rate requiring manual correction, which is a cumbersome task. In this thesis, we apply Out-Of-Distribution (OOD) methods to detect chemical compounds that are not informative for human scent analysis tasks. OOD detectors assume that there exists a distribution underlying the training classes samples, and detect samples not belonging to that distribution. We use the OOD detection to filter out chemical compounds that do not contribute to solving the gender classification, and identity verification tasks. We experiment with 8 OOD detection methods on: (i) chemical compounds dataset with manually checked labels of 70 compounds, $\approx$ 334 samples for each compound, and (ii) a single test measurement without ground-truth labels with $\approx$ 720000 samples. Given a predefined set of chemical compounds, which is small compared to the set of all compounds that may appear in a gas sample, we show that the OOD detection is a suitable method for filtering out chemical compounds not contributing to human scent analysis tasks.

Keywords: chemical compounds classification, chemical compounds OOD detection, GCxGC ToF mass spectrometry data analysis, near-OOD detection

Abstrakt

Analýza chemických sloučenin je zajímavý a důležitý problém [1, 2, 3]. Nedávno se ukázalo [4], že výstup z analytického chemického přístroje odhalující fyzikální vlastnosti plynu je vhodný pro úlohu klasifikace pohlaví a ověření identity. Správná klasifikace konkrétních chemických sloučenin přítomných ve vzorku plynu je pro uvedené úlohy klíčová. Klasifikace však do značné míry závisí na proprietárním softwaru s vysokou mírou chybovosti a vyžaduje ruční opravu, která představuje náročný úkol. V této práci se zabýváme klasifikací chemických sloučenin s pomocí detekce odchylek (anglicky 'OOD detection'). Tyto detektory předpokládají, že existuje rozdělení, které reprezentuje vzorek tréninkových tříd, a detekují vzorky, které do tohoto rozdělení nepatří. Detekci odchylek používáme k odfiltrování chemických sloučenin, které nepřispívají k řešení úloh klasifikace pohlaví a ověřování identity. Experimentujeme s metodami detekce na: (i) souboru chemických sloučenin s ručně ověřenými popisky 70 sloučenin, $\approx$ 334 vzorků pro každou sloučeninu, a (ii) testovacím měření bez popisků s $\approx$ 720000 vzorky. Vzhledem k předem definované množině chemických sloučenin, která je malá ve srovnání s množinou všech sloučenin, které se mohou objevit ve vzorku plynu, ukazujeme, že detekce odchylek je vhodnou metodou pro odfiltrování chemických sloučenin, které nepřispívají k řešení úkolu analýzy lidského pachu.

Klíčová slova: klasifikace chemických sloučenin, detekce odchylek chemických sloučenin, datová analýza hmotnostní spektrometrie GCxGC ToF

Contents

Acronyms

ASH extremely simple Activation SHaping.

AUC Area Under Curve.

AUPR Area Under Precision-Recall curve.

AUROC Area Under Receiver Operation Characteristic curve.

CNN Convolutional Neural Network.

DNN Deep Neural Network.

FN False Negative.

FP False Positive.

FPR False Positive Rate.

GC Gas Chromatography.

ID In-Distribution.

KL Kullback-Leibler.

kNN k-Nearest Neighbors.

ML Machine Learning.

MLE Maximum Likelihood Estimation.

MOOD Multi-level Out-Of-distribution Detection.

MS Mass Spectometry.

MSP Maximum Softmax Probability.

NMD Neural Mean Discrepancy.

NN Neural Network.

OE Outlier Exposure.

OOD Out-Of-Distribution.

PN Prior Network.

ReAct Rectified Activations.

ROC Receiver Operation Characteristic.

SVM Support Vector Machines.

TIC Total Ion Current.

TN True Negative.

ToF Time-of-Flight.

TOF-MS Time-of-Flight Mass Spectrometry.

TP True Positive.

TPR True Positive Rate.

VOC Volatile Organic Compound.

List of Figures

List of Tables

1 Introduction

Chemical compounds analysis is of great importance due to its wide area of application [1, 2, 3]. Recently, it has been shown [4] that a selected set of compounds found in human scent could be used to perform gender classification and identity verification.

A particular instrument of chemical analysis is a gas chromatograph, which, given a gas, produces a unique measurement, a spectrogram, reflecting the chemical composition of the gas. Current practice of chemical data analysis relies on proprietary black box software requiring human supervision, which is a tedious and prone-to-failure process.

In this thesis, we investigate ability of Out-Of-Distribution (OOD) methods to detect chemical compounds that were not identified as prospective for human scent analysis – compared to the set of all chemical compounds found in human scent, only 70 compounds were identified [4] as prospective. Our datasets consist of gas spectrogram measurements – spectrograms. In our first set of experiments, we artificially split our small labelled dataset to in/out-distribution sets in a compound-wise manner, and we measure performance of 8 OOD methods on this dataset. In the second set of experiments, we first train a classifier detecting 70 predefined compounds in an output of a gas chromatograph measurement consisting of $\approx 720,000$ spectra[1], and then we apply and evaluate OOD detection methods, reporting the performance by eyeballing.

OOD detection methods attempt to solve a known problem of (not exclusively) image classifiers[2] – they tend to give accurate predictions for known classes, and inaccurate, yet confident, predictions for objects not belonging to any of the classes the classifiers were trained on. For example, an image of a bear is always classified either as a dog, or as a cat, by a dog-cat classifier. *OOD detection* identifies unknown objects (classes) and either refuses to classify them or submits them for further evaluation through a 'manual annotation' performed by a human. Note that in our experiments, we apply image classification specific OOD detection algorithms due to theirs general applicability. Although the gas spectrograph output may be interpreted as a 2D image, investigation in that direction is left for a future research.

Given the source of our dataset, we feel entitled to mention specific challenges related to its origin. First, the output of the chromatograph is very sensitive to small changes in experimental setting, such as small variations of instrument temperature or wear of its internal parts. As a result, given the same compound, its spectra may vary considerably both in its quality, *i.e.* signal-to-noise ratio, and its time of detection since the beginning of an experiment, *i.e.* the retention time. Second, although the chromatograph produces a spectrum every 5 ms for a total duration of about one hour yielding $\approx 720,000$ spectra for each experiment, only 70 ground-truth labels were available to us from a single experiment.

[1] Experimental parameters of a gas chromatograph may be set differently depending on the purpose of a particular measurement. In the human scent analysis data available to us, the chromatograph was set to produce a single spectrum every 5 ms for the duration of $3,578$ s, yielding $715,600$ spectra.

[2] *Image classification* is the task of identifying objects (dog, cat, human) in images.

In the following list, we state the formal objectives of this thesis, and we summarize our contributions accordingly.

- **Familiarize yourself with the published work on the detection of outliers in: (i) image data, and (ii) gas chromatograph mass spectrometer data, if relevant literature is available.** We provide a comprehensive literature overview on OOD detection in image data in Sec. 2. In the best of our knowledge, there is no relevant literature available for OOD detection in gas chromatograph mass spectrometer data.
- **Review the literature on algorithms for chemical compound classification.** We provide a literature overview on chemical compound classification in Sec. 2.5.
- **Select and implement or develop a chemical compound classification algorithm, verify its function on data provided by the thesis supervisor.** We used a library SVM classifier and performed a series of quantitative experiments in Sec. 5.1.
- **Select and implement or develop at least two algorithms for out-of-distribution detection for use with the classifier from previous step, verify their function on data provided by the thesis supervisor.** We investigated eight OOD detection methods, and evaluated their performance with: (i) a qualitative experiment on a single gas chromatograph experiment in Sec. 5.2, and (ii) a quantitative experiment on a dataset consisting of artificially created in/out-distribution compounds in Sec. 5.1. In the quantitative experiments, we show that OOD detection is suitable for chemical compounds classification in our specific gas chromatography setting. Depending on the choice of artificially created in/out-distribution compounds, the performance varies significantly[3].

[3]The performance of detectors ranges from AUROC ≈ 76 with a 10-fold cross-validation to AUROC ≈ 94, depending on the relative sizes of in/out-distribution samples. Area Under Receiver Operation Characteristic curve (AUROC) is a metric used to evaluate performance of a classification model, where a perfect model has AUROC $= 100$.

2 Related literature

In this section, we describe existing OOD detection methods and, at the end, chemical compound classification. We begin with OOD detection. Many OOD detection methods have been developed. We divide these methods into multiple research areas and introduce representative methods for each of them. We propose a taxonomy of OOD methods based on techniques employed to derive OOD score.

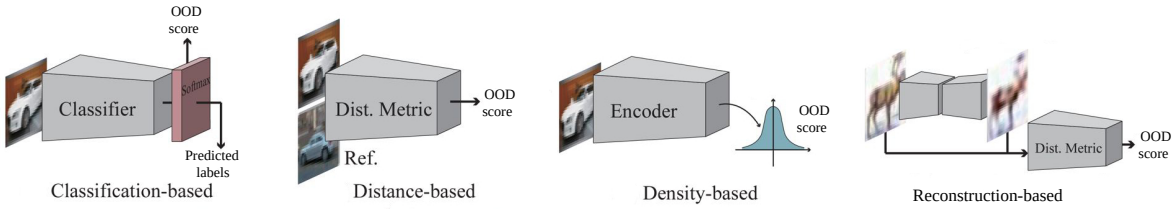


Figure 2.1: Comparison of OOD detection approaches, modified from [5].

We adopt the OOD detection taxonomy [6] with a following change – we add a subsection about zero-shot models as they do not fit into other categories in the original taxonomy. The taxonomy follows:

1. Classification-based methods
   (a) Output-based methods
   (b) Gradient-based methods
   (c) Bayesian methods
   (d) Methods using zero-shot pre-trained classifiers
2. Distance-based methods
3. Density-based methods
4. Reconstruction-based methods

In Fig. 2.1, we show how different OOD detection methods work in principle. We describe their limitations and requirements, as well as their key principle. Dividing methods into categories is important to quantitatively compare them.

For a test-time sample, an OOD detection method produces a single numerical value. An In-Distribution (ID) sample is expected to have a high value, and an OOD sample is expected to have a low value. In the following text, these values are referred to as OOD scores. OOD detection process is a threshold-based classification that produces binary output from OOD scores. If the score of a test-time sample is equal to or higher than the threshold, it is classified as ID. Conversely, if the score is below the threshold, it is classified as OOD.

2.1 Classification-based OOD detection

Classification-based methods derive OOD scores from a trained classifier.

2.1.1 Output-based methods

Some output-based methods use (i) custom training schemas, or (ii) extra OOD data to improve detection accuracy. To divide these methods, we adapt the following categorisation from [6]:

1. Post-hoc
2. Confidence enhancement
3. Outlier Exposure

Post-hoc methods use a classifier's output as-is. Confidence enhancement methods employ different training schemas. Outlier Exposure methods use additional OOD data samples for training.

Post-hoc methods take logits produced by a classifier and compute OOD scores by employing different transformations. There are two important design choices:

- scoring function - a function that transforms the classifier's logits to an OOD score
- aggregation function - an optional step where we aggregate logits after applying the scoring function to them

In 2017, Hendrycks et al. [7] introduced the Maximum Softmax Probability (MSP) detection method. **MSP** takes a data sample and feeds it to a trained classifier. It obtains softmax probabilities, takes the maximum from these values, and uses it as an OOD score for this sample. It is a straightforward method that was used to describe the OOD task and how the experiments should be conducted. This method is used as a reference point for comparison in the literature, is often denoted as the baseline method.

In 2017, Liang et al. [8] introduced **ODIN**, another simple method that builds on top of MSP and improves its performance with input pre-processing (adding small perturbations to input data points) and temperature scaling. The idea of temperature scaling was taken from [9]. It involves scaling the logits (inputs to the softmax function) by a temperature parameter T before computing the softmax function. It has the effect of sharpening the softmax output, making it easier to distinguish between in-distribution and out-of-distribution images. Temperature scaling was later implemented in other methods. The base version of ODIN does not use additional OOD data for fine-tuning. However, more recent versions of ODIN do so. If this is the case, ODIN is a representative of Outlier Exposure (OE) methods (which will be introduced later).

In 2020, Liu et al. [10] introduced **Energy-based detector** that uses energy as a scoring function. Unlike softmax confidence scores, energy scores are theoretically aligned with the probability density of inputs and are less susceptible to the overconfidence issue. The idea behind the so-called energy-based models (EBMs) [11] is to produce a scalar energy as a measure of compatibility with each configuration of the input variables. In this case, energy measures the compatibility of an image and its corresponding label. Small energy values correspond to highly compatible configurations of the variables, while large energy values correspond to highly incompatible configurations of the variables. Similarly to ODIN, energy might be used as a scoring function for a post-hoc method or as a trainable parameter that requires fine-tuning with OOD samples for a OE method.

In 2020, Sastry et al. [12] came up with **GRAM** that uses Gram's matrices to detect inconsistencies between the predicted class and the activity patterns of the classifier intermediate layers. In general, Gram matrices are used to compute pairwise feature correlations. In this work, they extended these matrices to characterize the activity patterns in the intermediate layers of the classifier for each predicted class. The intuition behind this is that if an image is predicted to contain a dog, but the classifier's activity patterns are atypical from the dog images that the classifier encountered during its training, then it is likely to be an OOD example. Before generating OOD scores, we compute: (i) feature correlations with Gram's matrices for every layer of the classification network and (ii) class-specific minimum and maximum values for the correlations of training images. Then we compute deviations of a test-time sample from training images. Here, a deviation is a number that indicates how far the tested value falls outside the range given by the minimum and maximum values of the training data. This deviations are computed layer-wise and summed to create an OOD score. Gram reaches state-of-the-art performance on two popular datasets: CIFAR-10 and CIFAR-100 [13]. It does not have any hyperparameters that need tuning. A drawback is the OOD detection speed, which is low.

In 2021, Sun et al. [14] introduced **Rectified Activations (ReAct)** detector. They compared the output of the penultimate layer (fully connected layer represented as an n-dimensional vector) between ID and OOD samples. The intuition behind this is that the variance for ID samples is more or less constant across the n dimensions, but it is quite large for OOD samples. ReAct refers to the process of removing any value that exceeds a certain constant at the second-to-last layer of a classifier. OOD score is extracted using the classifier with the clipped layer and a scoring function like MSP, ODIN or energy. It reaches state-of-the-art performance for some ImageNet-1k [15] benchmarks [4].

Multi-level Out-Of-distribution Detection (MOOD) was introduced in 2021 by Lin et al [16] as a novel framework that uses intermediate layer outputs rather than only using final layer outputs. Easy OOD examples are effectively detected early without propagating to deeper layers using coarse-level features like colour. However, more complex OOD samples will be detected in the deepest layers. The main advantage of MOOD is

[4]ImageNet-1k benchmark, refers to use ImageNet-1k dataset as ID dataset and evaluate with other datasets as OOD datasets, in the literature.

the complexity of inference, which is much lower than for other methods.

In 2022, Hendrycks et al [17] introduced two methods. **MaxLogit** gets the negative of a maximum from un-normalized logits as OOD scores. They emphasis that MSP and alike detectors do not perform well on large-scale datasets with a large number of classes that are possibly semantically very close to each other. MaxLogit outperforms MSP on a large-scale detection benchmark. The authors also introduce the **Kullback-Leibler (KL)** detector. This detector computes the KL divergence between un-normalized logits and a uniform distribution and uses it as OOD scores.

In 2022, Sun and Li [18] highlighted the problem with over-parameterized weight space of Neural Network (NN)s that leads to overconfident predictions. They propose to use sparsification on the weight space with their method **DICE**. The main idea is to rank weights based on a measure of contribution and select only the most contributing weights. The measure of contribution is computed with element-wise multiplication of a weight vector for a class and a feature vector of a test-time sample. A matrix of the measures is computed on ID dataset and the top k weights are selected based on the k largest elements in the matrix. The sparse weights matrix is used to propagate a test-time sample through the classifier to calculate softmax scores. Then, it utilizes the energy score to produce OOD scores. DICE is comparable with ReAct in the performance, but compromises classification performance (unline ReAct).

In 2022, Dong et al. [19] introduced **Neural Mean Discrepancy (NMD)** detector. This method takes channel-wise activation means between a test-time sample and training samples and computes their difference as element-wise subtraction. The difference results in a NMD vector that is passed to a binary classifier (logistic regression or multi-layer perceptron). The output of the classifier gives the OOD predictions. For the training samples, we get the neural mean vector from batch normalization layers. Thus, inference time is very small compared to other state-of-the-art methods. There are two main variants of the detector. The first has access to OOD data for training the classifier (OE). The second does not and uses artificial OOD examples by randomly permuting pixels of ID examples to train the classifier. They both reach state-of-the-art performance on CIFAR-10 benchmarks.

Recently, in 2023, Djurisic et al. [20] introduced a new post-hoc method **extremely simple Activation SHaping (ASH)** which is similar to DICE because it also uses sparsification. ASH, removes a large portion (*i.e.* 90%) of a sample's activations at a late layer, and the remaining portion (*i.e.* 10%) is simplified or adjusted (*i.e.* values are set at a constant value). These values are propagated forward to other layers, and the OOD score is computed with the energy score function. ASH has an interesting feature. It enables to select a custom 'ID-classification-performance-to-OOD-detection-performance' ratio depending on the specific requirements of the task at hand. ASH reaches state-of-the-art performance on some ImageNet-1k benchmarks together with ReAct.

Confidence enhancement methods change the training process of a classifier to account for OOD data. They do not require OOD samples for training. They change the loss

function or use feature engineering to generate augmented training samples. These methods do not reach the state-of-the-art performance of recent post-hoc methods,*i.e.* GRAM, but they improve performance of some methods (i.e. MSP, or Energy-based detectpr). Importantly, these concepts might be combined with other methods, leaving room for further improvement of state-of-the-art methods.

In 2018, Vyas et al. [21] used **ensemble of classifiers** where each is trained with a novel margin-based loss. To classify a test-time sample, first, the softmax probabilities of all classifiers are averaged, and the class with the highest average softmax score is chosen as the prediction. Then, for each of the classifiers, we compute both the maximum value and the negative entropy of the softmax probabilities (with temperature scaling). Finally, we compute the average of all these values to obtain the OOD detection score.

In 2019, Hein et al. [22] introduced **ACET**. They present an optimization for ReLU networks [5] through enhanced adversarial confidence training. ACET modifies the network directly via an adaptation of the training process so that uniform confidence predictions are enforced far away from the training data.

Another set of methods generates augmented samples to better distinguish OOD samples. Classifiers trained with these augmented data are used to predict the classes, and together with a post-hoc OOD scoring function, produce OOD score. The first method **MixUp** by Thulasidasan et al., 2020, [23] trains the classifier with additional samples that are generated during training by convexly combining random pairs of images and their associated labels. Similarly, Yun et al., 2019, [24] propose a different augmentation strategy **CutMix**, where augmented patches are cut and pasted into training images, where ground truth labels are also mixed proportionally to the area of the patches. The last and newest of this kind of methods is **PixMix** developed by Hendrycks et al., 2022, [25]. PixMix improves OOD detection simultaneously, as it improves classification accuracy. PixMix is comprised of two main components: (i) a set of structurally complex pictures ("Pix") and (ii) a pipeline for augmenting clean training pictures ("Mix"). At a high level, PixMix integrates diverse patterns from fractals and feature visualisations into the training set. As fractals and feature visualisations do not belong to any particular class, we train networks to classify augmented images as the original class, in standard data augmentation.

In 2020, **Generalized ODIN** was introduced by Hsu et al. [26], as an extension of ODIN [8]. It improves ODIN by modifying the original input pre-processing and re-training classifier with supervision on both class of ID samples and domain of ID samples.

In 2020, Meinke and Hein [27] introduced a method **Certified Certain Uncertainity(CCU)**. They focus on a so-called 'far-OOD' task, which means detecting OOD samples that are semantically far away from ID dataset. *I.e.* take SVHN [28] dataset with street view house numbers and CIFAR-10 dataset with objects like airplane, bird, truck, etc, for ID/OOD datasets respectively. CCU guarantees that a classifier produce low confidence for far-OOD task, it also computes upper bound for the worst-case scenario. CCU model

[5] ReLU networks refer to Neural Networks that utilize ReLU activation.

keeps high-confidence predictions in regions close to the training data, while it produces predictions close to uniform confidence for data points far away from the training. Their approach is based on a Maximum Likelihood Estimation (MLE). It uses Gaussian mixture models as density estimators. CCU reaches a near perfect score for the 'SVHN / CIFAR-10' (ID / OOD) benchmark.

In 2021, Macêdo and Ludermir [29] introduced **IsoMax**. It consists of using IsoMax loss (as a replacement for SoftMax) for training and an entropic score. IsoMax loss uses class prototypes, obtained from training data, and measures the distance between a sample's and prototype's high-level features with a performance enhancement similar to the temperature scaling. The entropic score computes the negative entropy of the classifier's output probabilities. Later in 2022, Macêdo and Ludermir [30] developed an enhanced version **IsoMax+**. The main changes were adding a distance scale and using the minimum distance score (minimum distance of a sample and a class prototype over all class prototypes) instead of the entropic one as the OOD detection score. The distance score is a trainable scalar parameter that is used to multiply distances. It performs slightly worse than GRAM [12], the best post-hoc method, but does not suffer from a low detection speed. The authors emphatize that speed is crucial for real-world large-scale applications.

In 2022, Wei et al. [31] introduced a novel loss LogitNorm that is used instead of the Cross-Entropy loss for classifier training. Their LogitNorm loss enforces a constant vector norm on the logits in training. It is based on the observation that even when most training examples are classified to their correct labels, the softmax cross-entropy loss continues to increase the magnitude of the logit vectors. A NN with LogitNorm improves performance of other post-hoc methods, *i.e.* MSP, ODIN, Energy, GradNorm.

Outlier Exposure methods represent methods that use auxiliary OOD samples in training [6]. Outlier Exposure generally leads to better performance, *i.e.* AUROC ≈ 99 on some popular benchmarks, compared to post-hoc and confidence enhancement methods. Outlier Exposure methods are suitable for applications where OOD detection score must be really high. However, great performance comes with the following problems: (i) collecting auxiliary OOD samples for training is not an easy task, as it is difficult to know a distribution of OOD samples in advanced, leading to high costs associated with collecting labeled OOD data, and at the same time (ii) measured performance is possibly overestimated by correlations between auxiliary OOD data, used for training, and real OOD samples, used for testing.

In 2018, Hendrycks et al. [32] first used an auxiliary OOD dataset to train a detector in the **Outlier Exposure** method. The auxiliary OOD samples are disjoint from test OOD samples. This paper provides a template for using OE in other Outlier Exposure [7]

[6] Note the difference between training and validation with OOD samples. *I.e.*, ODIN uses OOD samples as a validation set to fine-tune hyperparameters, where OE method train a network with OOD samples available.

[7] In the literature, Outlier Exposure is used for both the detection methods category and the specific

detectors. It requires to: (i) gather auxiliary OOD dataset, and (ii) define auxiliary OOD loss used to train the classifier. It significantly improves performance of MSP detector and it produces a perfect, AUROC = 100 classifier some benchmarks, representing far-OOD task. However, this method needs a large auxiliary OOD dataset, for example as much as ten times larger than the ID dataset, which is often not feasible for real datasets.

In 2020, Li and Vasconcelos [33] addressed the problem of the previous method by making a selection on auxiliary dataset, thus making it smaller. They implement different sampling strategies to produce the auxiliary dataset. They use so-called *adversial resampling* technique to obtain a set of OOD that is compact (small) and representative at the same time. The proposed technique leverages OOD loss to guide re-weighting of auxiliary dataset, encouraging the model to explore diverse examples that are challenging for OOD detection. This work gives a performance comparable to that of the previous Outlier Exposure method while having a much smaller auxiliary dataset. It results in a smaller computational and space complexity.

In 2020, Mohseni et al. [34] developed a classifier using self-supervised learning. They develop a classifier with two heads: (i) classification head using supervised learning with ID samples, and (ii) auxiliary head using self-supervised learning with OOD samples. An auxiliary dataset is unlabeled and disjoint from test dataset. After initial training of the main classification head on ID, the auxiliary head is trained with a mixture of labelled ID and unlabelled OOD samples. The classifier generates labels during the training process using a semi-random method.

In 2021, Papadopoulos et al. [35] introduced **Outlier Exposure with Confidence Control (OECC)** using a novel loss function. This loss function makes a classifier highly uncertain for OOD samples by producing a uniform distribution at the output of the softmax layer. They use OE training schema from [32] together with their loss function. OECC increase performance of GRAM [12] method mentioned before, making it a state-of-the-art method on CIFAR-10 benchmarks.

In 2021, Bitterwolf et al. [36] presented **Guaranteed Out-Of-distribution Detection (GOOD)** that concentrates on the worst-case scenario for OOD detection. In addition to the standard measure AUROC, they use so-called Guaranteed AUC (GAUC) for evaluation. This value represents AUC that is guaranteed in the worst-case scenario. They derive this value as an upper bound on a confidence level for a classifier using OOD auxiliary dataset. Similarly, they use confidence upper bound for loss function. The main contribution is the guarantee under the worst-case scenario.

In 2021, Yang et al. [37] introduced new so-called **Semantically Coherent Out-Of-distribution Detection(SC-OOD)** benchmarks and evaluation framework Unsupervised Dual Grouping (UDG). They point on the problems of some popular benchmarks, *i.e.* CIFAR-10 (ID) vs. TinyImageNet [38] (OOD). There are about 15% images from TinyImageNet with the same semantics as CIFAR-10 images. Both datasets contain images of

detection method.

cats and dogs with only a slight change (different breeds, skin color). Perfect performance on such benchmark results in non-realistic detector that will concentrate on the small covariance shifts too much and ignore semantics as a consequence. Their featured UDG framework process datasets to be used for semantically coherent detection. There are two steps: (i) pick out the ID classes from the OOD datasets and mark all images within these selected classes as ID samples, (ii) perform fine-grained filtering on images with wrongly-labeled categories (image of cat wearing a bow tie has bow time label). They use an unlabeled OOD dataset for training and filtering.

In 2022, Katz-Samuels et al. [39] introduced a framework WOODS that enables **using unlabeled 'wild' data** collected in deployment. When we introduced OE methods, we mentioned that performance may be overestimated by correlations between additional and real OOD samples. Unlabeled 'wild' data consists of both ID and OOD samples produced during the deployment of a OOD detector. The main idea is to solve a constrained optimization problem, by minimizing a detection error on an auxiliary OOD data while keeping the: (i) detection error on ID data below a threshold, and (ii) OOD detection error on ID data within certain limits. In their experiments, they simulate 'wild' data as a mixture of ID and OOD data. When the 'wild' data contains samples from test OOD dataset, the detector gives very good performance on CIFAR-100 benchmarks with large semantic shifts, *i.e.* far-OOD task.

In 2022, Ming et al. [40] introduced a framework **Posterior Sampling-based Outlier Mining (POEM)** that uses outlier mining. The goal of the outlier mining is the selection of samples from auxiliary dataset that are closer to the decision boundary between ID and OOD data. To collect OOD samples for training, it uses so-called Thompson sampling technique instead of a random sampling technique. It iteratively samples from the auxiliary OOD dataset based on a boundary score, where a high boundary score indicates sample being close to the decision boundary. This method has state-of-the-art performance on CIFAR-10 benchmarks and some CIFAR-100 benchmarks (together with WOODS).

2.1.2 Gradient-based methods

Previous, output-based methods utilize and an output space or a feature space of a classifier. Here, we describe a method that uses information from a gradient space instead.

In 2021, Huange et al. [41] introduced **GradNorm**. They compute KL divergence between: (i) a softmax output, and (ii) an uniform probability distribution. KL divergence [42] is a measure of a difference between two probability distributions and is defined as the expectation of the logarithmic difference between the model-predicted distribution and the reference distribution. ID data are expected to have a larger KL divergence because predictions tend to focus on one of the ground-truth classes, making it less uniformly distributed. This technique does not directly employ KL divergence, but instead utilises the gradient vector norm, which is propagated backwards from the KL divergence. GradNorm has state-of-the-art performance on one CIFAR-100 benchmark, but has significantly

worse performance, compared to state-of-the-art methods, on other popular benchmarks.

2.1.3 Bayesian methods

A **Bayesian model** is a type of statistical model that uses Bayes' theorem to derive posterior distributions based on prior distribution. These models use probability to represent all uncertainty within the model, both the uncertainty regarding the output (predicted classes) and the uncertainty regarding the input (image features) to the model. A **Bayesian neural network** is a NN that uses Bayesian formalism. Prior distribution is specified upon the parameters of NN and then, given the training data, the posterior distribution over the parameters is computed. Non-Bayesian methods (standard classifiers) train the network in a multi-task way, including both ID and OOD examples (for OE methods), to generate precise and consistent categorical predictions. However, these classifiers are not able to identify the source of a predictive uncertainty. Bayesian networks learn a distribution over weights (as in a standard classifier). *I.e.* a standard classification NN predicts which animal is in an image, whereas a Bayesian classification NN determines probabilities of each animal being in the image. A **Bayesian method**, in an OOD detection context, means using a Bayesian NN to derive uncertainty for the OOD detection. It is possible to convert a standard NN into a Bayesian NN to model the uncertainty. There are two limitations of Bayesian NNs: (i) they are computationally expensive, and (ii) it is difficult to acquire initial (prior) probabilities about the input. In the next paragraphs we introduce methods modelling uncertainty using approximations to overcome the problems.

Uncertainty in the model parameters leads to a range of possible predictive distributions for a given input data. The expected distribution, denoted as $\mathrm{P}\left(\omega_c \mid \boldsymbol{x}^*, \mathcal{D}\right)$, represents the average or most likely predictive distribution considering all possible values of the model parameters $\boldsymbol{\theta}$, given input data $\boldsymbol{x}^*$. To obtain the expected distribution it integrates or sums over all possible values of the model parameters:

$$\mathrm{P}\left(\omega_c \mid \boldsymbol{x}^*, \mathcal{D}\right) = \int \underbrace{\mathrm{P}\left(\omega_c \mid \boldsymbol{x}^*, \boldsymbol{\theta}\right)}_{\text{Data}} \underbrace{\mathrm{p}(\boldsymbol{\theta} \mid \mathcal{D})}_{\text{Model}} d\boldsymbol{\theta}, \tag{1}$$

where $\mathcal{D}$ is a dataset the model used for training and ω_c is a class label. The following Bayesian methods use the following equation:

$$\mathrm{p}(\boldsymbol{\theta} \mid \mathcal{D}) \approx \mathrm{q}(\boldsymbol{\theta}) \tag{2}$$

Obtaining true posterior distribution, denoted as $\mathrm{p}(\boldsymbol{\theta} \mid \mathcal{D})$, is computationally challenging, or impossible. Therefore, an approximation is necessary:

$$\mathrm{P}\left(\omega_c \mid \boldsymbol{x}^*, \mathcal{D}\right) \approx \frac{1}{M} \sum_{i=1}^{M} \mathrm{P}\left(\omega_c \mid \boldsymbol{x}^*, \boldsymbol{\theta}^{(i)}\right), \boldsymbol{\theta}^{(i)} \sim \mathrm{q}(\boldsymbol{\theta}) \tag{3}$$

Finding an exact solution to the integral in Eq. 1 is also computationally challenging or even impossible. Eq. 3 represents the approximation of the integral using sampling. Sampling involves randomly selecting values from a distribution to estimate the integral. This is used in **Monte-Carlo Dropout** [43], that uses dropout layers of a Deep Neural Network (DNN), or **Deep ensembles** [44], that uses ensembles of models, to produce uncertainty estimates. In Eq. 3 each $\mathrm{P}\left(\omega_c \mid \boldsymbol{x}^*, \boldsymbol{\theta}^{(i)}\right)$ in the ensemble represents a categorical distribution. A categorical distribution is a probability distribution that assigns probabilities to a set of discrete outcomes or categories. The categories are class labels y, and the probabilities represent the likelihood of each class label given the input $\boldsymbol{x}^*$ and a specific set of model parameters θ. The goal is to find a posterior model that is consistent in the region of training data (ID) and increasingly diverse when the input is far from the training data (OOD).

In 2018, Malinin and Gales [45] introduced a method that utilizes **Dirichlet Prior Networks (PNs)** for modelling uncertainty. There are three different types of uncertainty around the classification task: (i) *model uncertainty* that measures uncertainty in model parameters (weights) estimation, given training data, (ii) *data uncertainty*, that arises from the complexity of the data itself (due to noise, class overlap, etc.) and (iii) *distribution uncertainty*, or dataset shift, that occurs due to a mismatch between training and test distributions. Previous methods are not able to separate these uncertainties. Dirichlet PNs are explicitly constructed to capture both data uncertainty and distributional uncertainty. Distributional uncertainty is described by the distribution over predictive categoricals $\mathrm{p}\left(\boldsymbol{\mu} \mid \boldsymbol{x}^*, \boldsymbol{\theta}\right)$. Then the expected distribution is:

$$\mathrm{P}\left(\omega_c \mid \boldsymbol{x}^*, \mathcal{D}\right) = \iint \underbrace{\mathrm{p}\left(\omega_c \mid \boldsymbol{\mu}\right)}_{\text{Data}} \underbrace{\mathrm{p}\left(\boldsymbol{\mu} \mid \boldsymbol{x}^*, \boldsymbol{\theta}\right)}_{\text{Distributional}} \underbrace{\mathrm{p}(\boldsymbol{\theta} \mid \mathcal{D})}_{\text{Model}} d\boldsymbol{\mu} d\boldsymbol{\theta} \tag{4}$$

Dirichlet PNs model various types of uncertainty by producing distinct uni-modal Dirichlet distributions for ID samples and flat Dirichlet distributions for OOD samples. They use a loss function that incorporates KL-divergence between the model output and a target Dirichlet with a pre-defined precision value by utilizing it. There are more ways to measure the uncertainty: (i) taking probability of the predicted class and the entropy of the predictive distribution, (ii) compute mutual information between the categorical label and the model parameters. OOD scores are produced with a measure of uncertainty.

In 2021, Nandy et al. [46] introduced a method using Dirichlet PNs. The disadvantage of the previously mentioned method is that, for ID samples with high data uncertainties, the proposed loss function spreads out the target precision values among the overlapping classes, resulting in much flatter distributions. That leads to mis-classified samples. They suggest a novel loss that models the mean and the precision, of the output Dirichlet distributions, separately. It trains the Dirichlet Prior Network (PN) to generate flatter and more varied Dirichlet distributions for OOD samples.

In 2021, Kim et al. [47] introduced a method **Locally Most Powerful Bayesian Test**

(LMPBT). A Bayesian hypothesis test is a statistical method used to evaluate the strength of evidence for a particular hypothesis based on observed data. OOD detection is expressed as a Bayesian test, where the null hypothesis states that a test sample is an ID sample and the alternative hypothesis being that it is an OOD sample:

$$\begin{aligned} H_0 &: \theta = \theta_0, \\ H_1 &: \theta = \theta_1, \end{aligned} \tag{5}$$

Where $\theta = \theta_0, \theta = \theta_1$ are representative parameters of a deep generative model for ID samples and OOD samples respectively. Deep generative models consist of a generative model and an inference model, where the generative model defines the process of generating $\mathbf{x}$ from a latent variable $\mathbf{z}$, and the inference model infers the distribution of $\mathbf{x}$ given $\mathbf{z}$. Bayesian hypothesis test is based on the ratio of the posterior odds that the alternative hypothesis is true given the observed data:

$$\frac{P\left(H_1 \mid \mathbf{x}_t\right)}{P\left(H_0 \mid \mathbf{x}_t\right)} = \frac{L\left(\theta_1 \mid \mathbf{x}_t\right)}{L\left(\theta_0 \mid \mathbf{x}_t\right)} \times \frac{P\left(H_1\right)}{P\left(H_0\right)} \tag{6}$$

$P\left(H_i\right)$ is the prior probability for hypothesis H_i, and $L\left(\theta_1 \mid \mathbf{x}_t\right) / L\left(\theta_0 \mid \mathbf{x}_t\right)$ is the Bayes factor in favor of the alternative hypothesis, where $L\left(\theta_i \mid \mathbf{x}_t\right)$ are the likelihood of $\mathbf{x}_t$ under H_i. To complete the test, we must specify θ_0 and θ_1, and the prior probabilities for H_0 and H_1. θ_0 is specified as the MLE but it is difficult to estimate θ_1 for the lack of information on OOD samples. For that, an optimization task is solved for individual OOD samples to find θ_1. A computational approximation is used for the solution to decrease computational complexity. LMPBT performance is near perfect for some far-OOD benchmarks.

2.1.4 Methods using zero-shot pre-trained classifiers

The goal of **zero-shot OOD learning** is: (i) classifying test-time samples that belong to one of the previously seen (ID) classes, and (ii) detecting samples that are not part of any of the seen classes (OOD). Zero-shot models are trained on large datasets and demonstrate excellent performance on data they have not seen during training. Note that this area is also refered to as the 'zero-shot OOD detection'. **Zero-shot OOD detection** methods do not have acces to ID samples, therefore no closed-world (ID) classifier is built, and consequently the detection is based only on labels of ID classes.

In 2022, Ming et al. [48] introduced **Maximum Concept Matching (MCM)** that uses a distance function together with CLIP [49], a pre-trained language-vision model. CLIP is trained using contrastive learning on a large dataset, consisting of images and text caption pairs, allowing for a (zero-shot) transfer to various tasks, without the need of training a new Machine Learning (ML) model. The zero-shot classification is performed by comparing features from an image encoder to a set of text features from the text encoder, in a test-time image. The label, with the greatest similarity to the image, is the one that

is taken as a prediction. MCM computes the cosine similarity between image features of a test-time sample and the closest ID text prototype. The similarities are then scaled with softmax and a temperature scaling, similar to ODIN [8], and summed across all image classes to derive OOD scores.

In 2022, Esmaeilpour et al. [50] introduced **Zero-shot OOD detection based on CLIP (ZOC)** that also utilized pre-trained CLIP model. ZOC extends CLIP by training a textual description generator to produce OOD labels for testing samples (images). It performs OOD detection by comparing the similarity between a: (i) test-time image and ID labels, obtained from the pre-trained CLIP, and (ii) test-time image and candidate labels, obtained with the test-time image. The textual description generator is initialized and then fine-tuned on a validation dataset.

In 2023, Vojir et al. [51] introduced **Generic Representation based OOD detection approach (GROOD)** that uses a zero-shot classifier embeddings together with simple classifiers. Unlike MCM and ZOC, GROOD does not use a text encoder. Instead, it relies solely on the image encoder from CLIP or DIHT [52] models to extract a feature vector. The vectors are taken as input by two simple classifiers: (i) Liear Probe (LP) and (ii) Nearest Mean (NM). LP is a process that involves projecting data onto a line and then normalizing it using a softmax function. NM assigns data to the class with the closest mean. GROOD combines these two classifiers with the Neyman-Pearson task to further increase performance. It combines the output of LP and NM, trained on ID data, to model the distribution of the scores as a bivariate Gaussian. Then it formulates a two-class Neyman-Pearson task to calibrate the detector, *i.e.* set thresholds. The calibration involves setting a decision threshold such that ID classes are rejected evenly, without producing higher false negative rates for certain classes. GROOD gives state-of-the-art performance on the CIFAR-10 benchmarks.

2.2 Distance-based OOD detection methods

Distance-based methods are based on the idea that test-time OOD samples should be relatively far from the centroids, geometric centers, of ID classes.

In 2018, Lee et al. [53] came up with a detector that uses **Mahalanobis distance** and is compatible with any pre-trained softmax neural classifier. Previous methods explore mainly output space of classifiers [8], probabilities created by softmax. However, this method concentrates on a feature space of classifiers, outputs of deep layers. Mahalonobis detector fits pre-trained low-level and upper-level features by a class-conditional Gaussian distribution. To do this, one must first compute empirical class mean and class covariance of training (ID) samples. OOD detection score is calculated using the Mahalanobis distance with respect to the closest class conditional distribution (the class mean and covariance). Generally, the Mahalonobis distance calculates a distance between two data points using

[8] GradNorm utilizes a gradient space of a classifier.

empirical mean and covariance, where Euclidean distance uses only empirical class mean. It also uses input pre-processing for testing samples (adding small noise to input) for better ID and OOD separation.

In 2021, Ren et al. [54] published a paper that builds on the Mahalanobis detector and introduces so-called **relative Mahalanobis distance**. The method splits images into foreground and background and then calculates the Mahalanobis distance ratio between the two spaces. This method is hyperparameter-free, unlike the Mahalanobis distance method.

In 2021, Sehwag et al. [55] introduced **SSD**. They use self-supervised representation learning followed by a Mahalanobis distance in the feature space of a classifier. **Self-supervised representation learning** train a feature extractor by discriminating between individual instances from unlabeled data to learn a good set of representations. It attempts to bring each sample near its positive (similar) samples while pushing it away from negative (different) samples. This feature extractor is trained and used to extract feature vectors from the training ID dataset. The detector then evaluates the Mahalanobis distance between the features of a test-time sample and the mean and variance feature vector of the training set, similarly as in the previous distance-based methods.

In 2022, Sun et al. [56] introduced a method **Deep Nearest Neighbours** that uses the k-Nearest Neighbors (kNN) distance instead of the Mahalanobis distance. During testing, it takes a feature vector from a testing sample and normalize it. Then it computes the Euclidean distance between the feature vector of the test sample and each feature vector of ID samples. It sorts the ID feature vectors based on the distances in ascending order, and takes the first k distances and either compute a mean or a minimum, of these distances, to obtain an OOD score.

In 2022, Ming et al. [57] introduced **CIDER**. Similarly to SSD [55], it aims to create a hyperspherical embedding space in such a way that the extracted embeddings are more meaningful and understandable for OOD detection. Hyperspherical embeddings lie on the surface of a hypersphere. The main concept is to create a trainable loss function that optimizes two terms at the same time: (i) dispersion loss, that encourages larger angular distances between different class prototypes, (ii) compactness loss, that encourages samples to be close to the same class prototypes. A test-time input is predicted as OOD if it is significantly distant from the ID data in the embedding space. They use the kNN distance as a distance metric.

In 2022, Wang et al. [58] introduced **Virtual-logit Matching (ViM)** that utilizes information from both feature space and logits of a classifier. This method firstly computes a difference between a sample's feature vector and a principal subspace. Where the principal subspace represents the most important features of the ID data. The difference is then converted into an additional logit representing a virtual OOD class by projection and scaling. Finally, the softmax probability of the constructed OOD class is used as the OOD score. The smaller the original logits are, and the greater the difference is, the more likely it is to be OOD.

In 2023, Park et al. [59] introduced a method **NNGuide** that combines distance-based

and post-hoc methods. Post-hoc methods perform well on near-OOD detection task, *i.e.* detecting an unknown animal from ID images of cats and dogs. However, they are not as good in the far-OOD task, *i.e.* detecting a digit from ID images of cats and dogs. On the other hand, distance-based methods are better in far-OOD task and worse in near-OOD task. NNGuide balances the benefits and the weak spots of both. OOD score is a product of a 'base score' function, *i.e.* Energy [11] scoring function, and a 'guidance' term, a kNN distance used as in [56]. NNGuide uses a subset of the ID dataset to compute the distances and average over the nearest k of them. Importantly, this method is not only a standalone detector OOD, but is able to improve performance of other post-hoc and confidence-enhancement methods, *i.e.* MSP, GradNorm, Energy and Mahalanobis [7, 41, 10, 53].

2.3 Density-based OOD detection methods

Density-based method explicitly model ID by **probabilistic models** and find OOD samples as areas with low density during testing. Probabilistic models are a class of ML algorithms for making predictions based on the fundamental principles of probability and statistics. These models identify uncertainty in data and incorporate it into their predictions. These methods seem as a perfect candidate for OOD detection as they directly model the distribution. They have a similar performance in comparison with classification-based methods. But it is difficult and time-consuming to train and optimize them.

Density-based methods are sometimes referred to as generative models in some literature. These two terms are interchangeable. There is a difference between Bayesian methods and density-based methods. Density-based methods employ probability models, and Bayes methods, use Bayes models, in the context of OOD detection. A Bayesian model is a type of probabilistic model that uses Bayes' theorem to update beliefs or probabilities based on new evidence. A probabilistic method will learn the probability distribution over the set of classes and use that to make predictions (often using MLE). The main difference between a probabilistic model and a Bayesian model is the way they update probabilities. While a probabilistic model incorporates uncertainty, a Bayesian model specifically uses Bayes' theorem to update probabilities based on new evidence.

The Mahalanobis detector [53] that we characterized as a distance-based method is an example of density-based methods as well. When ID contains multiple classes, class-conditional Gaussian distribution are able to explicitly model the ID so that the OOD samples are identified based on their distances to the nearest class-conditional Gaussian distribution.

One group of methods that use probabilistic models are flow-based methods. In this paragraph, we will describe two such methods. In 2020, Zisselman and Tamar [60] introduced a residual flow, a novel flow architecture to learn the residual distribution from a base Gaussian distribution. They follow the success of the Mahalanobis detector [53] that fits mid-layers of a network to a Gaussian distribution. They fit mid-layers of a network and

use a more expressive density function, instead of a Gaussian, to model the ID. The density function is based on a deep normalizing flow. Normalizing flows are a popular model for high-dimensional data distributions in deep learning [61]. Main idea of normalizing flows is to model a target distribution as an invertible transformation of a base Gaussian distribution. Because the transformation is a bijection, it is possible to learn this transformation through MLE by training on ID data. Linear flow model establishes a relation between the MLE of a Gaussian model and a linear flow. The linear flow transformation is obtained analytically using the spectral decomposition of the empirical covariance matrix. Residual flow model extends the linear flow model to include non-linear components. The residual flow is applied to OOD detection by: (i) extracting mean feature vectors for each layer and for each class, (ii) finding the most probable class using the difference between training and testing feature vectors, and (iii) computing a confidence score as a difference between the feature vectors of a test sample and estimated mean. Small noise is also added to test-time samples, as in ODIN [8].

Theoretically, flow-based methods accurately estimate the probability of observing a given data point and being a perfect candidate for OOD detection. However, Nalisnick et al. [62] pointed on some problems of the flow-based methods. They showed that flows often assign a higher likelihood to OOD data than the ID data used for training of the detectors. This is true mainly for a far-OOD task. In 2020, Kirichenko et al. [63] followed this topic. They show that maximum likelihood objective has a limited impact on the ability of normalizing flows to detect OOD data. Instead, the inductive biases of the flow, *i.e.* modeling assumptions of the architecture, have a stronger influence on OOD detection. That is why change a structure of the flow layers to encourage the flow to learn the semantic structure of the target data rather than local pixel correlations, as with previous flow-based methods.

In 2020, Serrà et al. [64] discuss the drawbacks of generative models and how to solve them. They show that the most complex images tend to have the lowest probabilities, while the simplest images have the highest. They demonstrate this correlation by computing a quantitative estimate of complexity. This estimate of the complexity, provided by a PNG image compressor, is then used for OOD detection. OOD score is calculated by subtracting the negative log-likelihoods from the complexity estimate. The higher the score, the more the input resembles OOD which is opposite to the standard OOD detection definition. The score adjusts 'unusually' high probabilities and provides more accurate predictions for complex inputs. This method reaches near-perfect performance for near-OOD task and comparable performance with other density-based methods for far-OOD task.

In 2022, Yang et al. [65] introduced SEM method that directly models the density of semantic space. SEM score is used with a simple combination of density estimation in the low-level and high-level space. SEM has two distinct probability measures that are based on: (i) high-level features that contain both semantic and non-semantic data, and (ii) low-level feature statistics that only capture non-semantic image characteristics. By a simple combination, the non-semantic component is eliminated, leaving only the semantic

information. The base OOD score:

$$\mathrm{SEM}(\boldsymbol{x}) = \log p\left(\boldsymbol{x}_s\right) \tag{7}$$

Where $\boldsymbol{x}_s$ denotes features that only capture semantics. Probability $p\left(\boldsymbol{x}_s\right)$ is computed by a probabilistic model, such as a Gaussian mixture model using dense feature vectors. Eq. 7 is extended to the value of different probability measures based on low-level and high-level features as is explained above. This paper also introduces three Full-Spectrum OOD (FS-OOD) detection benchmarks that evaluate both semantic and covariate shift both on far-OOD and near-OOD tasks. This is done by creating a covariate-shifted ID dataset from an existing one and using it for a density estimation.

2.4 Reconstruction-based OOD detection methods

Reconstruction-based methods use trained reconstruction networks. Reconstruction-based NN trained with ID data usually leads to different outcomes for ID and OOD samples. OOD for detection is based on the difference between these outcomes. Reconstruction-based models with pixel-level comparison seem not a popular solution in OOD detection due to its expensive training cost. Models that use hidden features for reconstruction are more commonly employed.

Autoencoders are a type of NNs that consists of an encoder and a decoder. An encoder takes an input data and compresses it into a lower-dimensional representation, known as the latent space. The purpose of an encoder is to learn a compressed version of the input data that captures the most significant features or patterns. The decoder takes the compressed representation and attempts to recreate the original input data. The goal is to make the reconstructed data as close to the original input as possible. There is an asumption that the autoencoder is only able to effectively compress and reconstruct images that belong to the same class as the training data. OOD samples are expected to be difficult to compress and reconstruct accurately. Therefore, the reconstruction error is used as an OOD score.

In 2018, Denouden et al. [66] used **autoencoder** abilities together **with** the **Mahalanobis distance** to produce OOD scores. The results of their experiments concluded that methods that use reconstruction error as anOOD score are not able to detect certain anomalies that are distant from the known ID samples.

In 2022, Yang et al. [5] introduced **MoodCat**. Rather than reconstructing the entire image, MoodCat masks a random portion of the input image and identifies OOD samples by calculating a semantic difference between the original image and the synthesized one. MoodCat consists of an encoder and a decoder, where the encoder extracts low-level features and the decoder generates the synthetic part of image. ID samples are synthesized faithfully to their semantic meaning, while OOD samples highlight the mismatch between the input and the label. Masking is used to eliminate redundant information from the input image, allowing the generative model to concentrate on synthesizing the semantic meaning.

MoodCat's performance might be improved using outlier exposure technique, see Sec. 2.1.1. Altough MoodCat does not dependent on a classifier, it is designed to easily cooperate with one, as a 'plug-and-play' OOD detector.

2.5 Chemical compound classification

In this subsection, we provide a literature overview on chemical compounds classification. There are works [67, 68, 4], that concentrate on the chemical compounds classification from the chemical point of view. They use different measuring systems, *i.e.* gas chromatograph, and often proprietary software to process the output of a system's measurements. The works focus on the system configuration, rather that on the subsequent data analysis, which is our intention. There are not much works in chemical compound classification using data analysis, we only describe one work.

Croci et al. [69] used multi-dimensional gas chromatograph data to identify features of the background of a sample and attribute samples to a particular region or cultivar. A classifier is constructed to predict whether a test sample contains any compound or if it is just air, denoted as blanks. They treated the data as an image and apply kNN classification by the Wasserstein distance The Wasserstein distance is a measure of dissimilarity between probability distributions. The kNN separates the blanks with a perfect performance. However, for a more complicated problem, *i.e.* predicting from which cultivar is the sample from, the kNN classifier fails.

Clark [70], in his thesis, used different ML methods, *i.e.* SVM, kNN, Convolutional Neural Network (CNN), to classify one-dimensional chromatograph data. He concluded that using ML methods, for conventional gas chromatograph data, that is data similar to the data used in our experiments, is possible.

3 Datasets

In our experiments, we use mass spectra and retention times extracted from the output of a GCxGC-TOF-MS[9] system. Time-of-Flight Mass Spectrometry (TOF-MS) analysis produces mass spectra that are coming out of a comprehensive multi-dimensional Gas Chromatography (GC) (GCxGC) system every, *i.e.* 5 ms, for a total duration of 3600 s. For a single measurement, 2000 x 460 spectra are produced. The spectra are used for tasks related to the molecular profile of human skin Volatile Organic Compound (VOC)s. They are usually processed semi-automatically. Both by the proprietary black-box software ChromaTOF®, and by domain experts who manually check its output.

In this section, we describe the GCxGC-TOF-MS system and the data used in our experiments. There are four subsections in this section:

1. Gas Chromatography
2. Mass Spectometry
3. Multi-dimensional Gas Chromatography with Mass Detection
4. In-detail description of data used for experiments

The first three subsections discuss the main principles of measuring instruments, their components, and the purpose of these devices – this is based on a previous thesis [4]. These subsections are pre-requisities for the fourth subsection that describes the data used in our experiments (Sec. 5). The purpose of extracting this data was to analyze the molecular composition of the human skin scent with multi-dimensional Gas Chromatography in conjuction with Time-of-Flight Mass Spectometry (GCxGC-TOF-MS)

3.1 Gas Chromatography

Gas Chromatography is a simple physico-chemical separation technique that is used for subsequent GCxGC-TOF-MS analysis. It is suitable for separating and determining VOCs of various origins. The compound being analyzed is a gas or a liquid that has been converted into a gas, and the results obtained are retention times and m/z (mass/charge) ratios.

This technique works by separating individual sample molecules via a mobile phase, usually an inert 'carrier' gas, and a stationary phase, which is placed in the chromatographic column. The stationary phase interacts with components of a sample that are carried away by the mobile phase, delaying their movement based on chemical interactions. Therefore, different molecules will move through the column at different speeds based on their relatedness to the stationary phase [71].

The **gas chromatograph** (visualized in Fig. 3.1) consists of a: (i) pressure cylinder with carrier gas and regulator, (ii) sample injection, (iii) heated column compartment and

[9]We denoted the GCxGC-TOF-MS system as the 'chromatograph' in the previous sections.

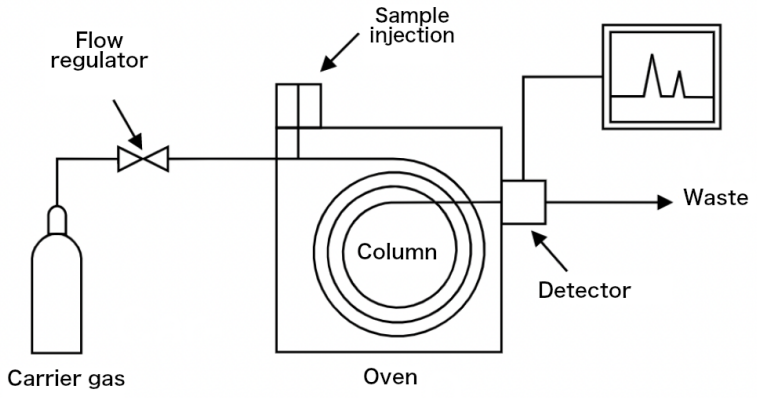


Figure 3.1: Gas chromatography block diagram. Taken from [4].

(iv) detector. In this subsection, we provide a short explanation of these components. A sample, in liquid or gas form, is injected into the carrier gas stream in an injection chamber. It is then gasified (if needed) and carried to the column where the separation process occurs. Individual components of the sample flow to the detector where they are detected. A response from the detector is recorded as a chromatogram [72].

3.1.1 Mobile phase

The main role of the **carrier gas** (mobile phase) is to carry molecules of a sample throughout a column. Because the carrier gas must not interact with individual sample components, various inert or non-reactive gases such as helium, nitrogen, argon, or hydrogen are used.

3.1.2 Stationary phase

The stationary phase is the most important factor in choosing a chromatographic column. This selection determines the final resolution of the column and simultaneously influences other sampling parameters.

3.1.3 Sample injection

In this stage, a sample is introduced to the GC system. There are different techniques for the injection process. A common technique involves directly injecting a liquid sample into the column.

3.1.4 Detectors

In Gas Chromatography, a variety of detectors are used, and the choice of a detector depends primarily on the application and the compounds being analyzed. A detector captures the physico-chemical properties of analytes eluting from a column and provides a response, which is then amplified and converted into an electronic signal, resulting in a chromatogram. The dataset from our experiments uses a GC with TOF-MS detector for measurements. We dedicate the next subsection to that.

3.2 Mass Spectometry

A combination of Gas Chromatography with Mass Spectometry (MS) (GC-MS) is a standard choice for a detector that offers low detection limits without requiring perfect separation of all components in a sample. It provides maximum information with minimal sample quantity. MS operates on the principle of ion separation in vacuum in the gas state. It is determined by the ***ratio of ion mass to charge (m/z)***. Molecules that have been separated by chromatography are ionized. The internal energy of these ions is too high, causing them to break apart. These charged fragments are then guided and accelerated into a mass analyzer, where they are classified according to their m/z values. Distinct signals from fragments with varying m/z ratios allow for the subsequent detection and identification of individual compounds [73].

3.2.1 Analyzers

Analyzers distinguish individual ions according to different m/z values. A Time-of-Flight (ToF) analyzer was used to generate the dataset used for our experiments.

Time-of-Flight analyzer offers a high resolution and the highest scanning speed among common analyzers, making them ideal for integration with multidimensional gas chromatography techniques.

3.3 Multi-dimensional Gas Chromatography with Mass Detection

In the analysis of volatile compounds in complex real samples, GC often fails as a separation tool [74]. An effective way to improve separation is to utilise multiple separation mechanisms within a single analysis. **Comprehensive two-dimensional gas chromatography (GC×GC)** offers up to several orders of magnitude greater peak capacity than conventional chromatographic methods. The separation capacity is expressed as $n_1 * n_2$ where n_1 is the peak capacity of the first column and n_2 is the peak capacity of the second column [75].

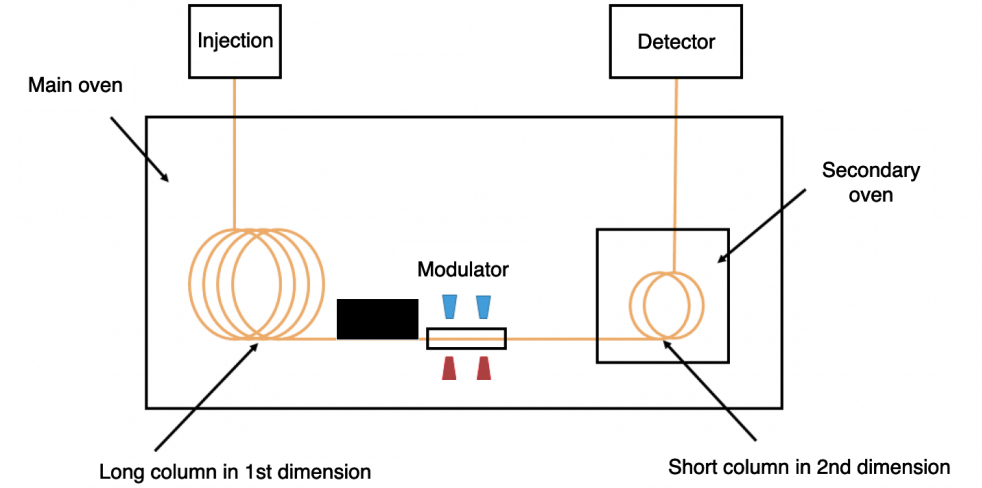


Figure 3.2: GC×GC block diagram with separate ovens and two-stage modulation. Taken from [4].

3.3.1 Separation

In GC×GC, an entire sample is separated by two columns with different stationary phases. One of them is non-polar and the other moderately to strongly polar. This allows for different methods of division. By utilizing chromatographic columns with different selectivity, it is possible to ensure that the various compounds have independent (uncorrelated) retention times in both dimensions [76, 77]. See Fig. 3.2 for one possible GC×GC system layout.

3.3.2 Modulation

For effective GC×GC separation, it is crucial to ensure continuous injection of the sample into the second column to avoid compromising the separation achieved in the first column. This is achieved using a modulator, which captures a small fraction of the effluent from the first column, refocuses it, and rapidly introduces it into the second column. The modulation time along with the separation time in the second dimension is known as the **modulation period** and typically ranges from 1 to 12 seconds [78]. The GCxGC-TOF-MS system that was used to collect the data used in our experiments uses a cryogenic modulator, with modulation period 10s, that is illustrated in Fig. 3.3.

3.3.3 Detection

GC×GC is commonly combined with a Mass Spectometry as it provides qualitative and structural information about the monitored analyte, serving as an additional third

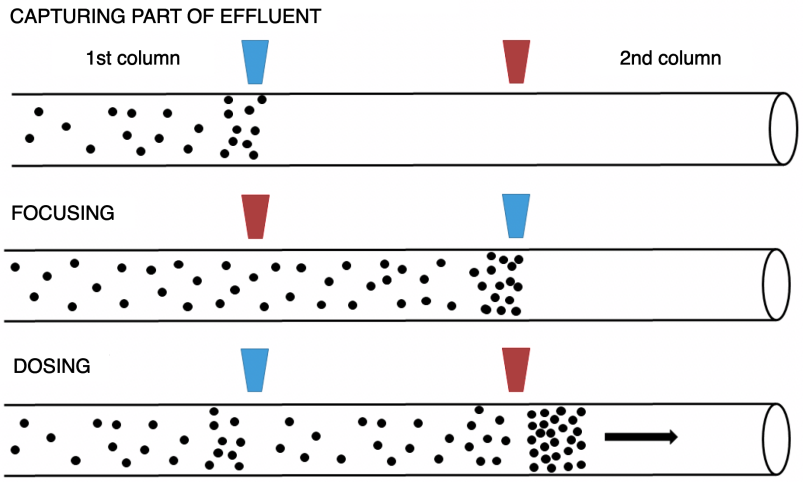


Figure 3.3: The concept of the two-stage cryogenic modulation. It involves utilizing a heating jet represented by the red nozzle and a cooling jet represented by the blue nozzle. Taken from [4].

dimension in multidimensional separation. **TOF-MS** is the most popular analyzer used in combination with GC×GC. TOF-MS is used for the system that gathers the data used for our experiments.

3.3.4 2D chromatograph

Output from GC×GC is a broad series of conventional chromatograms from the second dimension, which are transformed and stacked side by side using software to create a two-dimensional chromatogram (see Fig. 3.4, step 2). One dimension is represented by **retention time** from the first column. The second dimension represents the retention time from the second column. Visualization of the 2D chromatogram (see Fig. 3.4, step 3) is typically done by representing peaks in a contour plot, using colors, shading, or contours to indicate signal intensity. Occasionally, 3D visualizations are used for presentation purposes [75].

3.4 In-detail description of data used for experiments

3.4.1 Data creation

The data were created from human skin samples via a GCxGC-TOF-MS as part of a previous thesis [4]. The thesis studied the molecular profile of the human skin scent samples for gender classification and compound identification. This could potentially be used for scent analysis in forensic practice by building an extensive international database

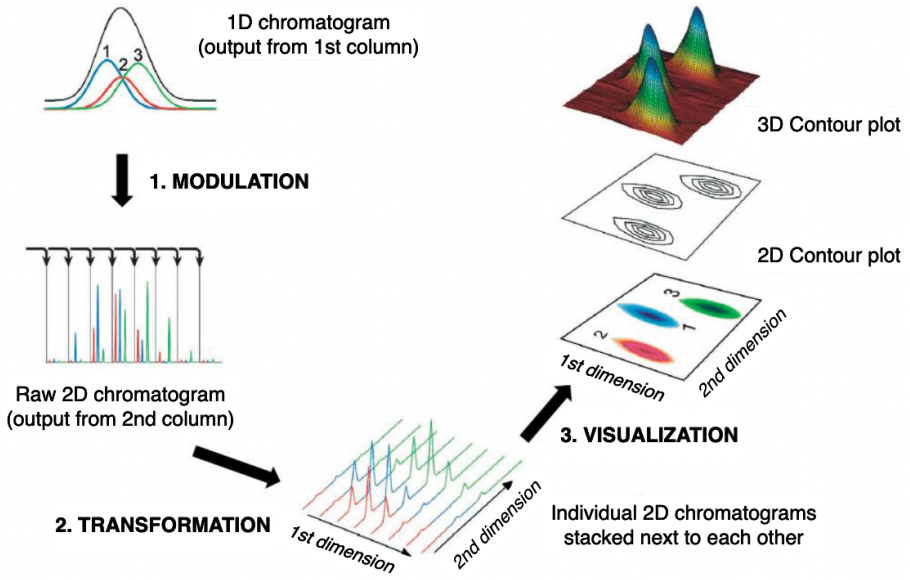


Figure 3.4: Diagram of GC×GC chromatogram formation and visualization. Taken from [4].

of scent samples and creating accurate classification models. The author of the thesis states that analyzing VOCs in complex samples such as human scent, traditional GC may not be an effective separation technique. Therefore, they used a multi-dimensional GCxGC-TOF-MS.

Chromatographic analysis

The measured data were first processed with the ChromaTOF®proprietary software. In chromatogram analysis, compounds whose main peak reached a signal-to-noise ratio (S/N) value greater than 500 were considered. The identification of compounds was based on the comparison of their mass spectra with the mass spectrum library *NIST MS Search 2.0*. After an initial manual evaluation of 20 samples, **70 compounds were identified in almost all** human skin scent **samples**. Note that while each GCxGC-TOF-MS measurement has 2000 x 460 spectra in total, only 70 of these are identified in all measurements. These compounds were automatically searched in the chromatograms by the *ChromaTOF®*software and then manually checked. Total Ion Current (TIC) chromatogram represents the summed intensity in the entire range of m/z values detected at every point (retention times in the first and second dimension) in the analysis. In Fig. 3.6, a TIC chromatogram is shown. The output of the GCxGC-TOF-MS chromatographic analysis are spectra for each sample (measurement) that are difficult to distinguish from one another based on their retention time. The resulting spectra from the system are one-dimensional and represent the mass-to-charge ratios (m/z) and their relative frequency (Fig. 3.5). These spectra are used as an input for our classifier.

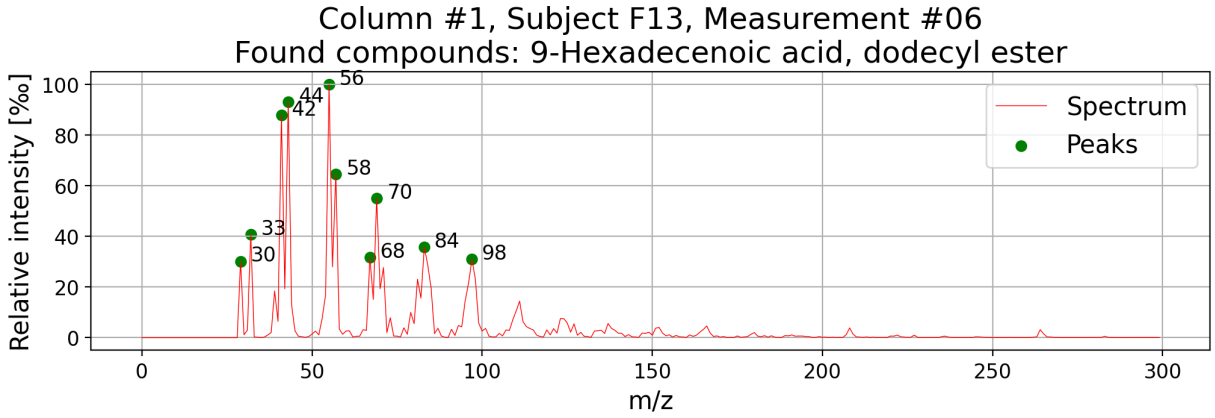


Figure 3.5: Mass spectrum of a dodecyl ester of 9-Hexadecenioc acid, with its mass-to-charge ratios on the x-axis and its relative intensity compared to the highest spectrum peak on the y-axis. The skin scent was sampled from a female subject number 13 with measurement number 6, executed on column number 1. Top 10 highest peaks are shown as green dots with their respective m/z value. Only spectra with m/z ratio values up to 300 were retained due to the absence of non-null values after the cutoff.

3.4.2 Compounds dataset

The Compounds dataset contains manually curated mass spectra obtained from 31 GCxGC-TOF-MS human scent samples (measurements) and contains 70 labelled compounds. The dataset was collected for the task of gender classification and contains an approximately equal number of spectra for each of the 70 compounds.

Data description

The dataset contains **23,430 samples** where each sample $\mathbf{x} \in \mathbb{R}^{801}$ and the first 29 dimensions are equal to zero. The dataset is made up of human skin scent from 40 individuals, with an equal number of males and females (20 each). The dataset include **70 labelled compounds** (as described in the last subsection). These are the classes used in our experiments. 62% of all samples are taken from female individuals, 32% are taken from male individuals. In the dataset, we have samples produced by three different settings of GCxGC-TOF-MS columns. As a result, samples from different settings have different retention times and levels of noise. We have between 300 and 350 spectra per compound for the majority of compounds, with the exception of four compounds which have a lower count, specifically 284, 208, 172, and 127.

Retention times

In this dataset, the spectra from multiple GCxGC-TOF-MS chromatograms were manually checked and labelled. Each spectrum contains the 801-dimensional vector as specified

above. It also contains retention times of the respective GCxGC-TOF-MS chromatogram. Retention times are measured in seconds. In our experiments, we build two variants of a SVM model where they differe in the usage of retention times. The first model that does not utilize retention times takes the training dataset where each sample $\mathbf{x} \in \mathbb{R}^{801}$. The second model that utilize retention times takes the training dataset where each sample $\mathbf{x} \in \mathbb{R}^{803}$.

Ground truth data format description
For each sample, the connected metadata are included in the sample filename. The metadata include description of the spectra, we refer to them as ground truth in the context of ML. Here is a sample ground truth with description (the sample's spectrum is visualized in Fig. 3.5):

`system_1_F1_01_Caffeine`: sample's filename

- `system_1`: The number of GCxGC-TOF-MS system configuration (There are 3 in total).
- `F13`: Represents a volunteer's unique code ($F1, \ldots, F20$ for female and $M1, \ldots, M20$ for male).
- `06`: The order of the individual's measurement $(01, \ldots, 31)$
- `9-Hexadecenoic acid, dodecyl ester`: Represents an annotated compound found in the sample.

Collecting training data for the detectors setup
Distance-based OOD methods require data from the training split of the Compounds data to compute distances during inference time. We save **data points** $\mathbf{x} \in \mathbb{R}^{803}$ when including retention times, or $\mathbf{x} \in \mathbb{R}^{801}$ otherwise. The **data labels** $y_i \in \{0, \ldots, 69\}$ associated with the data points are also saved. Note that the validation split of the Compounds dataset is used for experiments and thus cannot be used for the score generation. We take 20% of the training split of the final model (from cross-validation) for the quantitative experiments. And we take 100% of the training split of the same model for the qualitative experiments.

3.4.3 Test dataset

The Test dataset is one GCxGC-TOF-MS measurement where the **ground truth of individual mass spectra is unknown**.

Data description

The Test dataset is a GCxGC-TOF-MS chromatogram with 460 different retention times t_1 on x-axis and 2000 retention times t_2 on y-axis. See Fig. 3.6, where we visualize the Test dataset as a TIC chromatogram by summing across all m/z values. Each point in the chromatogram represents a single mass spectrum $\mathbf{x} \in \mathbb{R}^{801}$. We denote each spectrum as a testing sample. In order to deploy a classification model trained on the Compounds dataset, we transpose and flatten the data. Then the Test dataset has 920,000 samples where each sample $\mathbf{x} \in \mathbb{R}^{801}$. We mask all samples that have zero spectra, as the machine does not generate data for the entire duration of a measurement. Then the Test dataset has 715,600 samples where each sample $\mathbf{x} \in \mathbb{R}^{801}$.

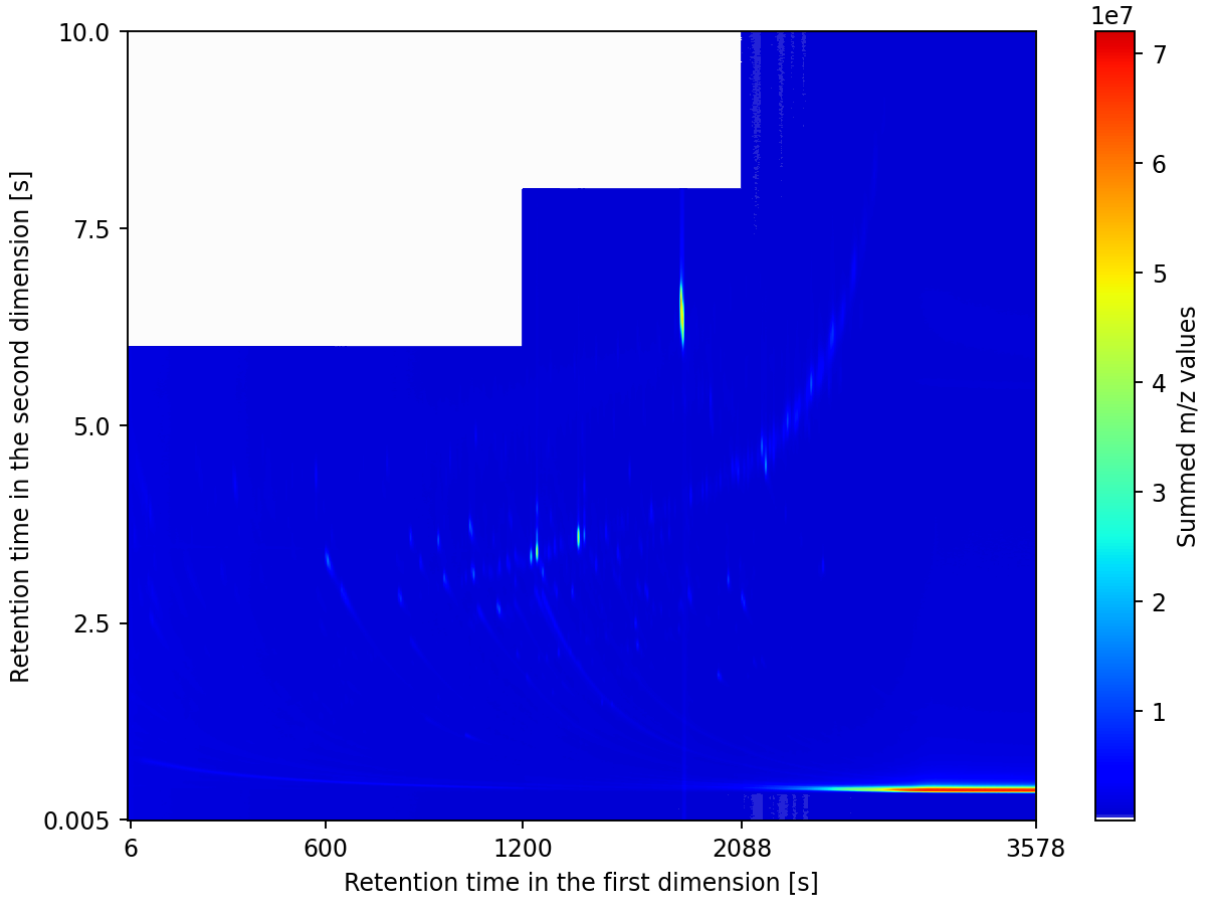


Figure 3.6: Test dataset sample, Total Ion Current (TIC) chromatogram of a sample of human skin scent. Each point in the image is a sum of all mass spectrum values belonging to the same scan.

Retention times

Similarly as with the Compounds dataset, we have two datasets that differ in the usage

of retention times for our model. Consequently, a test sample $\mathbf{x} \in \mathbb{R}^{801}$ or $\mathbf{x} \in \mathbb{R}^{803}$. We need to calculate retention times according to the GCxGC-TOF-MS system configuration. The system adjusts the measuring periods depending on an arrangement of the columns. *I.e.* a modulation period, the period during which a sample is injected into the second column, is 10 s. However, for the system, it is known that measuring for more than 6 seconds in the initial phase is not meaningful. So the chromatograph is set up to only produce outputs for 6 s (not 10 s). This differs throughout the process, and is visualized in the chromatogram[10] with white color. We calculate retention times[11] as follows:

Column index	Retention time (t_1)
0 - 199	$t_1 = 6 \times (\text{column index} + 1)$
200 - 310	$t_1 = 6 \times 200 + 8 \times (\text{column index} + 1)$
311 - 459	$t_1 = 6 \times 200 + 8 \times 111 + 10 \times (\text{column index} + 1)$

Row index	Retention time (t_2)
0 - 1999	$t_2 = 0.005 \times (\text{row index} + 1)$

3.4.4 Data normalization

We normalize spectra from both datasets with the L^{Max} norm, sometimes also called the infinity norm (L^{inf}). Max norm scales each value in the 801-dimensional vector by its maximum absolute values. If x is the spectrum vector, then the normalized spectrum vector is $y = x/z$. Infinity norm $\mathrm{L}^{\mathrm{Max}}$ is defined as:

$$\mathrm{L}^{\mathrm{Max}} : z = \|x\|_\infty = \max |x_i| \tag{8}$$

We scale the retention times as follows:

$$t_1 = \frac{t_1}{3600}$$
$$t_2 = \frac{t_2}{10}$$

The constants were selected according to the measurement systems found in the data.

[10] We also see the retention times in the first dimension are not uniform.

[11] We compute the total duration of the experiment with the computed retention times: 3578 s = 6 s $*$ 200 + 8 s $*$ 111 + 10 s $*$ 149. TOF-MS produces spectra every 5 ms. We compute the total number of spectra for the Test dataset as the total duration of the experiment multiplied by the TOF-MS measuring period, both in seconds: 715600 = 3578 $\div$ 0.005

4 Methods

In this section, we describe an SVM classifier, followed by description of investigated OOD detection methods. Since the OOD detection methods were developed for image data, we use their modified versions that suit our case. The classifier-based OOD detection methods require a classification model that is pre-trained on some labelled dataset. In our case, we train an SVM on the training subset of the Compounds dataset. An overview of the OOD detection is shown in Fig. 4.1.

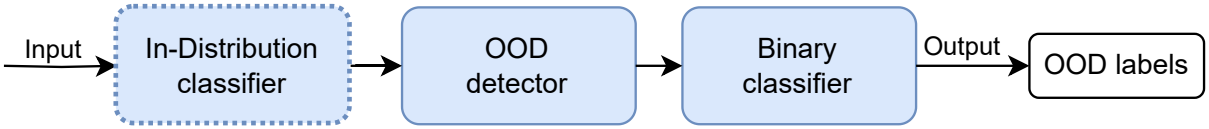
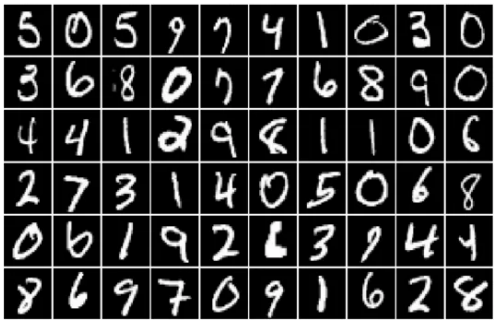


Figure 4.1: High-level OOD detection overview. An OOD detector takes the Compounds or Test datasets, optionally processed by a ID classifier, to produce OOD scores. A binary classifier then takes the scores as input and returns OOD labels. The labels denote whether an input is In-Distribution or Out-Of-Distribution.

Note that distance-based OOD detection methods do not require a pre-trained classification model to predict OOD labels. They predict the labels based on the provided data and a distance function.

4.1 Out-Of-Distribution Problem Statement

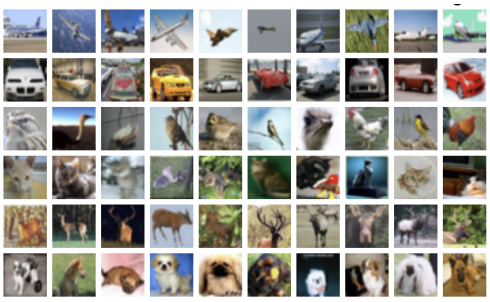
Out-Of-Distribution detection is a process of identifying data samples that are from a different distribution than the distribution the ML model was trained on. ID is the distribution of the training data and OOD represents data samples with *covariance shift* or *semantic shift*. An instance of covariance shift occurs when comparing images captured by a camera and animated images. Semantic shift refers to the existence of a class that is not present in the ID domain. Example of covariance and semantic shifts are shown in Fig. 4.2, 4.3.



(a) MNIST contains images of handwritten digits

(b) SVHN contains images of street view house numbers

Figure 4.2: An example of a covariance shift between images from MNIST [79] dataset and SVHN [28] dataset. Covariate shift occurs when the distribution of input data in a ML model training and testing data differs. The semantics of the datasets are identical as they both have images of digits. But their domain is different: (a) handwritten digits, and (b) street view digits.



(a) CIFAR-10 contains images 10 objects: airplane, automobile, bird, cat, deer, dog, frog, horse, ship, trucks

(b) SVHN contains images of street view house numbers

Figure 4.3: An example of a semantic shift between images from CIFAR-10 [13] dataset and SVHN [28] dataset. Images of SVHN are street view digits, images of CIFAR-10 are animals or vehicles.

OOD detection is a binary classification task. Let $\mathcal{D}_{in}$, $\mathcal{D}_{out}$ denote two distinct data distributions. We call $\mathcal{D}_{in}$ **In-Distribution** and $\mathcal{D}_{out}$ **Out-Of-Distribution**. Consider a ML model f trained on a dataset drawn from the distribution $\mathcal{D}_{in}$. In test-time, we deploy f on another dataset that drawn from a mix of $\mathcal{D}_{in}$ and $\mathcal{D}_{out}$ distributions. OOD detector G predicts whether a test-time sample $\mathbf{x}$ is In-Distribution or Out-Of-Distribution:

$$G(\mathbf{x}, f) = \begin{cases} 1 & \text{if } \mathbf{x} \sim \mathcal{D}_{in} \\ 0 & \text{if } \mathbf{x} \sim \mathcal{D}_{out} \end{cases} \tag{9}$$

4.2 Multi-Class Classification with Support Vector Machines

In this subsection, we describe the classification model. This is the first step in the OOD detection. In the classification task, the goal is to obtain a classifier that is trained on the Compounds dataset. This trained classifier is used in the next steps of OOD detection.

The output of this step is to obtain: (i) ID classification performance on the Compounds dataset, and (ii) logits extracted from the SVM classifier. **SVM's logits** are used by classification-based OOD detectors that are described in the following subsection. **ID classification performance** is reported as part of the OOD detection evaluation. ID classification denotes the classification of the Compounds data.

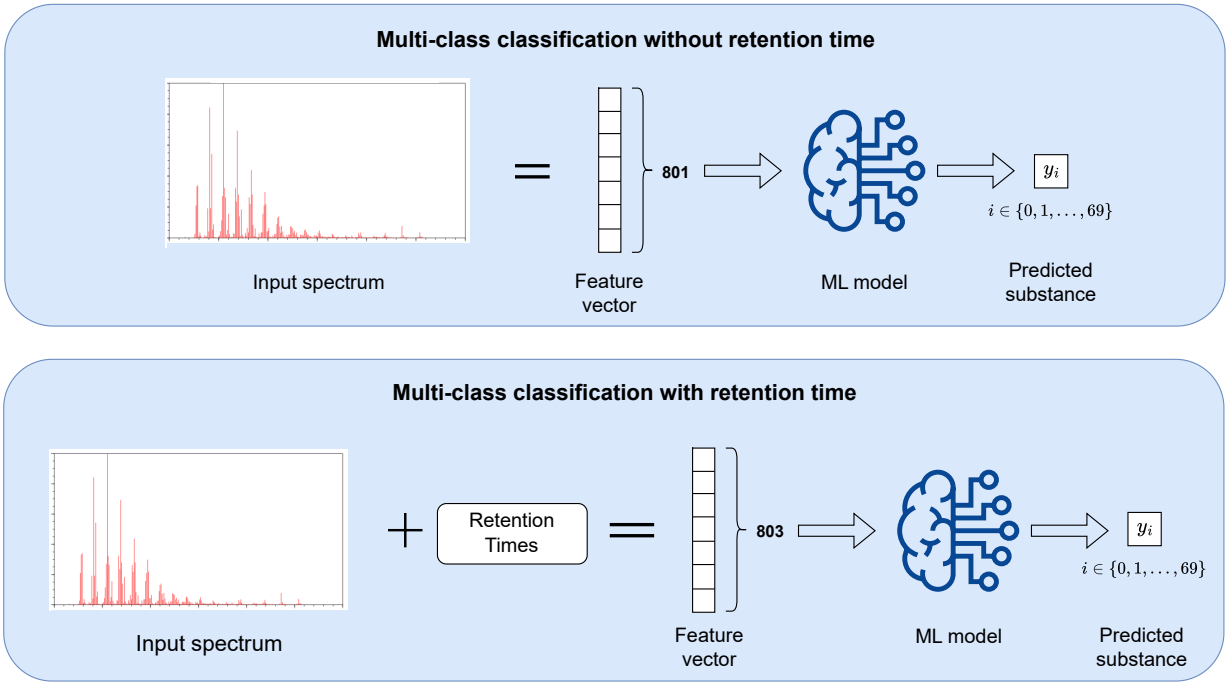


Figure 4.4: Multi-class classification for a SVM model, top: without, bottom: with retention times available. Machine Learning model takes a mass spectrum, optionally extended by retention times, as the input and predicts one of 70 compounds.

Multi-class classification is an instance of a supervised learning problem in which each input sample can be associated with a single label that indicates which class it belongs to. In our case, we assign one of the 70 known classes to a GCxGC-TOF-MS input. The input is an 801-dimensional or 803-dimensional vector. In Fig. 4.4 is a graphical representation of the task.

Before we introduce the method we used in the experiments, SVM, we outline other methods for the problem. A simple classifier can be created with a **dot product**. We compute the dot products between a test-time data vector and all data vectors from the training

dataset. This results in a vector of dot products from which we select the highest value. From that we take the associated label from at the index of the highest value.
k-Nearest Neighbors (kNN) first normalizes the data and obtains k nearest neighbors. For a test-time sample it assigns the class label that is the most frequent among the k neighbors.
Based on the Bayes theorem, the **Naive Bayes Classifier** gives the conditional probability of an event A given an event B. During training, it calculates class probabilities and conditional probabilities for each of the values of a spectrum and retention times. In prediction, it applies Bayes' theorem and iterates through all features to get the most probable class label for a test-time sample.
These methods (and possible more) are easily interpreted, easy to implement, and do not require complex training schemes. But they fail to capture complex patterns in the data. In the Compounds dataset the spectra for the same class (compound) differ quite significantly and require more complex methods.

CNN is the foundation of all classification-based OOD methods. It would require no modification to the model in order to run the methods. Building CNN for our data is technically possible with a small network. CNN in general are known to capture more complex patterns in the data which is also needed. But given the Compounds dataset size, it is very difficult to train a CNN that would: (i) not overfit the dataset, and (ii) have sufficient classification performance. We choose not to use CNN either.

SVM select the *optimal* separating hyperplane (decision boundary) that separates a n-dimensional space (801 or 803 in our case) into distinct classes. This decision boundary is called a **hyperplane**. Support Vector Machines (SVM) are based on the concept of maximizing the margin. Where the margin is the minimum distance from the separating hyperplane to the closest data point from the training dataset. SVM is expected to cope well with noise or more complex data patterns. We trained a CNN model and found that SVM model is superior in both classification and OOD detection performance without the need for complex fine-tuning. SVMs are also easier to implement, easier to interpret and require less computation. For these reasons, we select the SVM. More details on the SVM are given in Sec. 4.2.2.

4.2.1 Overall approach

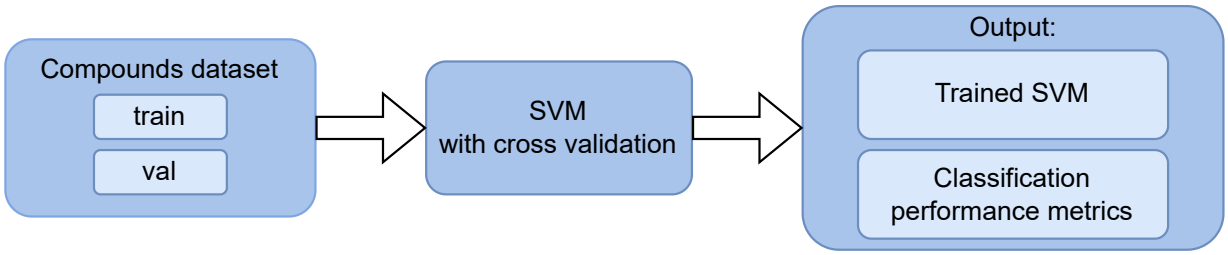


Figure 4.5: In-Distribution classification overview. SVM model is trained on the Compounds dataset divided into train/validation split by a cross-validation. The trained model and its performance metrics are taken as output for later OOD detection.

We train a multi-class classifier to classify GCxGC-TOF-MS. The input is either a 801-dimensional vector or a 803-dimensional vector. In the second case, the last two dimensions represent GCxGC-TOF-MS retention times. The output is a single label $y_i \in \mathcal{Y} = \{0 \dots 69\}$ where $\mathcal{Y}$ denotes a set of 70 known substances (classes). We implement a Support Vector Machines (SVM) model for this task. An overview of the classification process is visualized in Fig. 4.5.

4.2.2 Architecture

Model variants
We obtain both mass spectra and retention times from a GCxGC-TOF-MS chromatogram. We train two models: (i) SVM **without** access to retention times, and (ii) SVM **with** access to retention time. These models differ by the input's dimensions. The input is a: (i) 801-dimensional vector, or (ii) 803-dimensional vector respectively. We train both models with the same configuration. We report their performance in the Sec. 5.

Support Vector Machines optimization task
Let us formulate the SVM model as an optimization task. For a training dataset $\mathcal{T}$:

$$\mathcal{T} = \{(\mathbf{x}_i, y_i)\}_{i=1}^{m} \quad \text{, where } \mathbf{x}_i \in \mathbb{R}^n,\ y_i \in \{-1, 1\}, \tag{10}$$

where m is the number of samples in $\mathcal{T}$.

The optimization task, also known as the primary task, is formulated as:

$$\begin{aligned} \mathbf{w}^*, b^*, \xi_i^* = \arg\min_{\mathbf{w},b,\xi_i} \left(\frac{1}{2}\mathbf{w}^T\mathbf{w} + C\sum_{i=1}^{m}\xi_i \right) \\ \text{subject to } \mathbf{w}^T\mathbf{x}_i + b \geq +1 - \xi_i, \text{when} \quad y_i = +1 \\ \mathbf{w}^T\mathbf{x}_i + b \leq -1 + \xi_i, \text{when} \quad y_i = -1 \\ \xi_i \geq 0, \forall i \end{aligned} \tag{11}$$

Where $\mathbf{w}^*$, b^*, ξ_i^* are optimization variables that SVM solves and C is a model's hyper-parameter that can be fine-tuned. The first term in the optimization task maximizes the **margin (width) of a linear decision boundary** $\frac{2}{\mathbf{w}^T\mathbf{w}}$ by actually minimizing $\frac{1}{2}\mathbf{w}^T\mathbf{w}$ where $\mathbf{w}^T\mathbf{w}$ denotes a dot product. The decision boundary is a line in 2D feature space, but is generalizable to higher dimensions. The minimisation task involves the addition of **slack variables** ξ_i, which are summed. The variables allow that some training points may be mis-classified. A regularization parameter C is used to regulate this. SVM that enables for mis-classification is called a **soft-margin classification**. Due to the possible high dimensionality of the vector variable $\mathbf{w}$, we solve the dual problem of the optimization task. This is described in detail in [80].

'One-vs-all' classification strategy

We employ the **'one-vs-all' classification strategy** to adapt a binary SVM for a multi-class classification. There are several strategies for reducing a multi-class classification to a multiple binary classification. We use the so-called **'one-vs-all' strategy** that trains 70 distinct classifiers. Each of them is a binary classifier that predicts whether a sample belongs to the class $y_i \in \mathcal{Y} = \{0 \dots 69\}$ or if it belong to the rest of classes $\mathcal{Y} \setminus y_i$.

Probability estimates

For the purpose of OOD detection, the model must produce predictions that represent a probability distribution. SVM makes its predictions using a decision function that does not follow a probability distribution. That is, their values do not add up to one. We compute probability estimates as an extension to the SVM model. We use the method from [81] implemented in the sklearn library. This uses an improved implementation of the **Platt scaling** [82].

The goal of the probabilistic estimation is to calculate the likelihood of a data point $\boldsymbol{x}$ belonging to the class i:

$$p_i = P(y = i \mid \boldsymbol{x}), \quad i = 1, \dots, k, \tag{12}$$

where k is the number of classes. We estimate the likelihoods as r_{ij} with **pairwise ('one-vs-one') class probabilities**:

$$r_{ij} \approx P(y = i \mid y = i \text{ or } j, \boldsymbol{x}) \approx \frac{1}{1 + e^{A\ \hat{f}+B}} \tag{13}$$

Where $\hat{f}$ is a **decision value** for $\boldsymbol{x}$. Decision values are calculated by summing the contributions from the support vectors over all combinations of two classes. Although this technique is referred to as a 'one-vs-one' (or pairwise) approach, it can also be used with our 'one-vs-rest' classifier. The parameters A and B are estimated by minimizing the negative log likelihood of the training data with a 5-fold cross-validation.

After collecting all r_{ij} values, there are several proposed approaches to obtain $p_i, \forall i$ [81]. We use one such approach and solve the following optimization problem:

$$\begin{aligned} \min_{\boldsymbol{p}} \quad & \frac{1}{2} \sum_{i=1}^{k} \sum_{j:j \neq i} (r_{ji} p_i - r_{ij} p_j)^2 \\ \text{subject to} \quad & p_i \geq 0, \forall i, \quad \sum_{i=1}^{k} p_i = 1 \end{aligned} \tag{14}$$

The implementation [80] guarantees convergence to the unique optimum of this problem.

4.2.3 Training details

In the implementation, we use the Python programming language. And we use the *sklearn.svm* Python library for the implementation of the SVM. Specifically, we use the *sklearn.svm.SVC* function. This library is internally implemented with the *libsvm* library [80].

Training parameters
We train the SVM model with the SVC module from the sklearn library. We use the following parameters:

```
SVC(kernel=KERNEL_FUNCTION,
    gamma=GAMMA,
    C=C,
    decision_function_shape='ovr',
    probability=True,
    break_ties=True,
    class_weight='balanced')
```

We consider `KERNEL_FUNCTION`, `GAMMA` and `C` as the hyper-parameters that are fine-tuned. `decision_function_shape='ovr'` means we use the 'one-vs-rest' transformation. `probability=True` means that we have to create probabilistic estimates for our model. `break_ties=True` means the model break ties according to the confidence values.

The model consists of 70 'one-vs-rest' models and we define different regularization parameters C_i for each class i. As we set the argument `class_weight='balanced'`, we set the weights for the regularization parameter C based on the counts of a particular class. We do this to account for small class imbalances of the Compounds dataset.

Stratified 5-fold Cross-Validation
We implement a 5-fold cross-validation to train the models. **K-fold cross-validation** is a technique used to evaluate the performance of ML models. It splits the dataset into k subsets of the same size. Then the model is trained and evaluated k times using a different subset as the validation set each time. That is, it uses 4 subsets as training set and 1 subset as the validation set. We prefer this sampling technique over the conventional 'train-test' split as we have greater confidence in the model's performance on unseen data and it is easier to detect when the model is overfitting. **Stratified k-fold cross-validation** is a k-fold cross-validation where each divided subset has an approximately equal proportion of samples from each target class as the entire set.

Obtaining final model
For the experiments, we need a trained classifier. We get it with the cross-validation. For each fold, we evaluate the model and save its performance. We choose the model with the best validation performance as the final model.

Grid Search to fine-tune hyper-parameters
The implementation of the SVM uses these hyper-parameters: (i) `KERNEL_FUNCTION`, (ii) optional parameter `GAMMA` associated with the kernel function, and (iii) a regularization parameter `C`. We implement a simple exhaustive method where we iterate over all triplets (`KERNEL_FUNCTION`, `GAMMA`, `C`) to obtain the best model. We train the model using a triplet of hyper-parameters and evaluate it with a 5-fold cross-validation to obtain performance metrics. After searching through all possible triplets, we select the triplet that gives the greatest performance.

The best model was trained with the linear kernel and `C` = 1000. The large value of `C` implies a lower number of support vectors and enforcing a 'hard margin classification'. We use these parameters for all experiments.

4.3 Used OOD detection methods

In this subsection, we describe the OOD detection methods that are used in the experiments. Each method generates OOD scores that are essential for the next step of the OOD detection. The input for a detection method is: (i) a data sample we want to classify (ii) the Compounds dataset and (iii) a trained SVM classifier obtained from the previous step (Sec. 4.2). Whether we use the trained classifier or some statistics derived from the Compounds dataset depends on a specific OOD detector. The resulting score is a single value that is computed for every input data point.

We adapted eight different OOD detectors for the task:

- MSP detector [7]

Detector name	Scoring function	Aggregate function	Distance function
MSP	Softmax	Max	–
MaxLogit	–	Max	–
KL	KL divergence		–
Energy-based	Exponential	LogSum	–
Mahalanobis	–	–	Mahalanobis
KNN	–	–	kNN
SSD	–	–	Mahalanobis
NNGuide	Exponential	LogSum	kNN

Table 4.1: Attributes of detectors. Where a function is not defined it is labeled as '–'. KL divergence is a function that combines scoring and aggregate functions.

- MaxLogit detector [17]
- KL detector [17]
- Energy-based detector [10]
- Mahalanobis detector [53]
- Deep Nearest Neighbours detector (KNN) [56]
- SSD detector [55]
- NNGuide detector [59]

In this subsection, we demonstrate how each of them generates OOD scores. The implementation of these methods is taken from the public code available in the code repositories of the respective methods. Part of the evaluation script was taken from the NNGuide [59] public repository. It was modified as follows. Distance-based methods take data points as input (spectra + retention times) instead of feature vectors. We modified the scaling process for the Mahalanobis and SSD score. A public repository with our implementation is available on GitHub [12].

Following the taxonomy defined in Sec. 2 we divide this subsection into: (i) post-hoc methods, and (ii) distance-based methods in our method. The third part subsection introduces NNGuide, a detector that represents both post-hoc and distance-based methods.

There are three parameters that determine an OOD detection method: (i) scoring function, (ii) aggregate function, and (iii) distance function. Scoring and aggregate functions are parameters for the post-hoc detectors and distance function is a parameter for the distance-based methods. In Tab. 4.1, we list used parameters for all methods.

[12] https://github.com/lindepav/OOD_detection_for_GCxGC_classification_task

4.3.1 Post-hoc methods

Overall approach

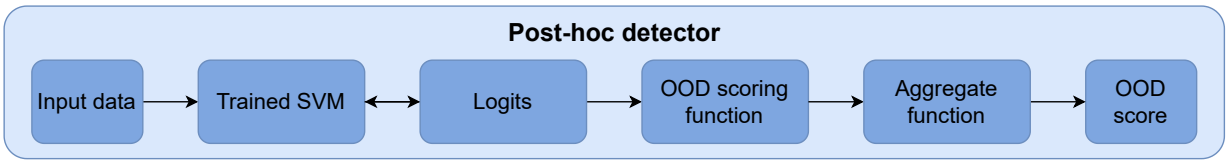


Figure 4.6: Out-Of-Distribution detection score generation overview for post-hoc methods. Post-hoc methods produce scores from logits extracted from a trained SVM model. OOD scoring function and aggregate function are used to derive the OOD score.

MSP, MaxLogit, KL and Energy-based detectors are instances of post-hoc output-based methods. An illustration of the process of creating OOD scores for post-hoc methods is presented in Fig. 4.6. These methods differ based on the choice of an: (i) **OOD scoring function** and (ii) **aggregate function**. The following paragraphs explain how a method calculates the OOD score. The logits are used as an input for all post-hoc methods. We explain the process of obtaining logits from the SVM model next.

Extracting logits from the SVM model

Logits are input for a post-hoc scoring function. The original methods require logits obtained from a CNN. Therefore, we need to make minor modifications to the implementation. We extract the output probability estimates from the SVM model (that was trained on the Compounds dataset) and obtain the **logits** as is described in Algorithm 1.

Algorithm 1 Convert SVM Probability Estimates to Logits

Require: *probs* : Prediction probabilities from SVM probability estimates
Ensure: *logits* : Logits calculated from probabilities
eps ← a small constant
probs ← $\max(\min(\text{probs}, 1 - \text{eps}), \text{eps})$ ▷ Clip probabilities for numerical stability
logits ← $\log\left(\frac{\text{probs}}{1-\text{probs}}\right)$ ▷ Convert probabilities to logits

MSP detector

Maximum Softmax Probability (MSP) [7] takes a data sample (mass spectrum + retention times) and feeds it to trained SVM. It calculates logits and uses softmax to obtain class probabilities. Then it takes the maximum and uses it as an OOD score for this sample.

$$\mathrm{MSP}(\mathbf{x}) = \max_i \left(\frac{\exp\left(f(\mathbf{x}_i)\right)}{\sum_{j \neq i} \exp\left(\mathbf{x}_j\right)} \right), \text{ where } i, j = \{0, \ldots, 69\} \tag{15}$$

MaxLogit detector
MaxLogit [17] takes the negative of a maximum from the logits as an OOD score.

$$\mathrm{MaxLogit}(\mathbf{x}) = \max_c \left(f(\mathbf{x}_c)\right), \text{ where } c = \{0, \ldots, 69\} \tag{16}$$

KL detector
KL detector [17] computes the Kullback-Leibler divergence between the probability distribution estimate obtained from logits $q(\mathbf{x}_c)$ and a generated uniform distribution $p(\mathbf{x}_c)$. In general, the KL divergence quantifies how much one distribution differs from another. In the implementation, we compute the KL divergence as the cross-entropy loss (without reductions) between the logits and the generated uniform distribution. The reason for that is that the computation of cross-entropy loss is less demanding. The cross-entropy is the sum of the entropy $H(p(\mathbf{x}))$ and the KL divergence between $p(\mathbf{x})$ and $q(\mathbf{x})$. The entropy term $H(p(\mathbf{x}))$ is constant for each c since the target is fixed and its distribution never changes. Therefore, we neglect this term.

$$\mathrm{KL}(\mathbf{x}) = D_{\mathrm{KL}}(p(\mathbf{x}) \| q(\mathbf{x})) = \sum_{c=0}^{69} p(\mathbf{x}_c) \log \frac{p(\mathbf{x}_c)}{q(\mathbf{x}_c)} = -\sum_{c=0}^{69} q(\mathbf{x}_c) \log q(\mathbf{x}_c) \tag{17}$$

Energy-based detector
Energy-based detector [10] uses energy as a scoring function. Unlike softmax confidence scores, energy scores are theoretically aligned with the probability density of inputs and are less susceptible to the overconfidence issue. Energy measures the compatibility of a data point and its corresponding label.

Energy-based detector computes a score for test-time data $\mathbf{x}$ with their extracted logits $f(\mathbf{x})$ using the energy function:

$$\begin{aligned} \mathrm{Energy}(\mathbf{x}, T) &= T \cdot \log \sum_{c=0}^{69} \exp(\frac{f(\mathbf{x}_c)}{T}) \\ \mathrm{Energy}(\mathbf{x}, T = 1) &= \log \sum_{c=0}^{69} \exp(f(\mathbf{x}_c)) \end{aligned} \tag{18}$$

In our experiments, we use the temperature $T = 1$.

4.3.2 Distance-based methods

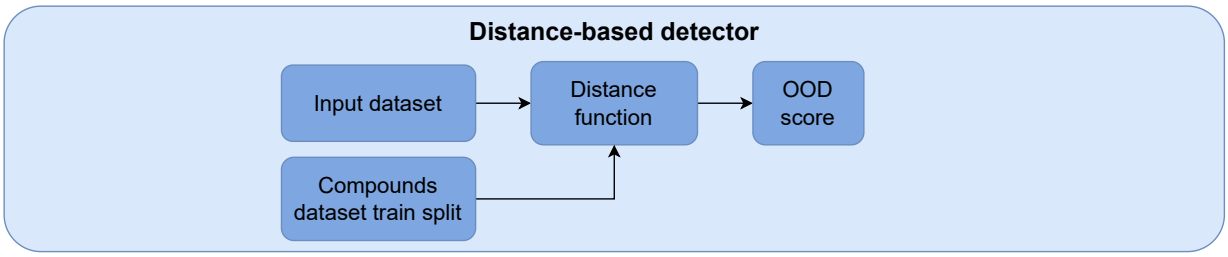


Figure 4.7: Out-Of-Distribution detection score generation overview for distance-based methods. Distance-based detectors measure distances between input data and reference training split of the Compounds dataset with a distance function.

Mahalanobis, Deep Nearest Neighbours and SSD are instances of distance-based methods. An illustration of the process of creating OOD scores for distance-based methods is presented in Fig. 4.7. These methods differ mainly by the selection of the distance function. We provide the definitions of the distance function in the following subsections.

For the methods, we load the data points $\mathbf{x}^{train} \in \mathbb{R}^{803}$ or $\mathbf{x}^{train} \in \mathbb{R}^{801}$ and the data labels $y^{train} \in \{0, \dots, 69\}$ from the training split of the Compounds dataset. How we save these is described in Sec. 3.4.2.

Mahalanobis detector

Mahalanobis [53] fits pre-trained low-level and upper-level features by a class-conditional Gaussian distribution. We modify this to use the unprocessed data points spectra instead (mass spectra + retention times) as SVM does not produce these features. Mahalanobis also uses input pre-processing for testing samples (adding small noise to input) for better ID and OOD separation. We did not achieve better performance with this, so we omitted it.

Prior to calculating the Mahalanobis score, we first calculate the empirical class mean and covariance of the ID training data $\{(\mathbf{x}_1^{train}, y_1^{train}), \dots, (\mathbf{x}_N^{train}, y_N^{train})\}$:

$$\begin{aligned} \widehat{\mu}_i &= \frac{1}{N_i} \sum_{j:y_j=i} \mathbf{x}_j^{train}, \text{where } i = \{0, \dots, 69\}, N_i = \text{occurence of class i} \\ \widehat{\mathbf{\Sigma}} &= \frac{1}{N} \sum_{i=0}^{69} \sum_{j:y_j=i} \left(\mathbf{x}_j^{train} - \widehat{\mu}_i\right) \left(\mathbf{x}_j^{train} - \widehat{\mu}_i\right)^\top, \text{where } N = \text{total number of samples} \end{aligned} \tag{19}$$

For a test-time data point $\mathbf{x}$ we compute the Mahalanobis distance as follows:

$$\text{Mahalanobis'}(\mathbf{x}) = \min_c \left(\left(\mathbf{x} - \widehat{\mu}_c\right)^\top \widehat{\mathbf{\Sigma}}^{-1} \left(\mathbf{x} - \widehat{\mu}_c\right) \right), \text{where } c = \{0, \dots, 69\} \tag{20}$$

Mahalanobis score can be derived by taking the negative value of the computed distance. In the implementation, we first normalize the scores as follows:

$$\text{Mahalanobis}(\mathbf{x}) = \exp\left(-\left(\text{Mahalanobis'}(\mathbf{x})/\text{M}\right)/2\right), \text{ where M = number of classes} \quad (21)$$

Deep Nearest Neighbours (KNN) detector

Deep Nearest Neighbours [56] uses kNN distance instead of Mahalanobis distance. First we find k-nearest neighbors for a test-time data point $\mathbf{x}$ with inner product distances $\langle ., . \rangle$. The neighbors are computed from the Compounds training split data points. We obtain only the top k nearest training data points. The KNN score is computed as follows:

$$\begin{gathered} \text{KNN}\left(\mathbf{x}, k\right) = -\min_{j}\left(\langle \mathbf{x}, \mathbf{x}_{(j)}^{train} \rangle\right), \\ \text{where } j = \{1, \ldots, k\}, \text{ and } \mathbf{x}_{(j)}^{train} \text{ is j-th nearest neighbor} \end{gathered} \quad (22)$$

We use the Faiss library that has an efficient implementation of quering the nearest neighbors. In our experiments, we use $k = 10$.

SSD detector

SSD [55] detector first calculates the empirical class mean and covariance of the Compounds dataset training split similarly to the Mahalanobis detector. However, here we use only the data points $\{\mathbf{x}_1^{train}, \ldots, \mathbf{x}_m^{train}\}$ without labels. We set the labels with k-means clustering to $y_1, \ldots y_m$. Where m is a method's parameter. If $k = 1$, we produce artificial labels to 'classify' all training data into one class. In other words, we do not use any clustering. We compute the class mean and covariance of the ID training data $\{(\mathbf{x}_1^{train}, y_1), \ldots, (\mathbf{x}_N^{train}, y_N)\}$ as defined in 19.

For a test-time data point $\mathbf{x}$ we compute the SSD score as follows:

$$\begin{gathered} \text{SSD}\left(\mathbf{x}, m\right) = \min_{c}\left(\left(\mathbf{x} - \widehat{\mu}_c\right)^{\top} \widehat{\mathbf{\Sigma}}_c^{-1} \left(\mathbf{x} - \widehat{\mu}_c\right)\right), \text{where } c = \{0, \ldots, m\} \\ \text{SSD}\left(\mathbf{x}, m = 1\right) = \left(\mathbf{x} - \widehat{\mu}\right)^{\top} \widehat{\mathbf{\Sigma}}^{-1} \left(\mathbf{x} - \widehat{\mu}\right), \\ \text{where } \mu, \Sigma^{-1} \text{ are mean and inverse covariance of the whole Compounds data} \end{gathered} \quad (23)$$

We use the same normalization as defined in Eq. 21. In our experiments, we use $m = 1$.

4.3.3 NNGuide detector

NNGuide [59] combines a distance-based score (KNN) with a post-hoc score (Energy). An illustration of the process of creating OOD scores for NNGuide is presented in Fig. 4.8. Logits and data points from Compounds training split are extracted in the same way as for post-hoc and distance-based methods respectively. In our experiments, we use the temperature $T = 1$ and $k = 10$ for kNN distance.

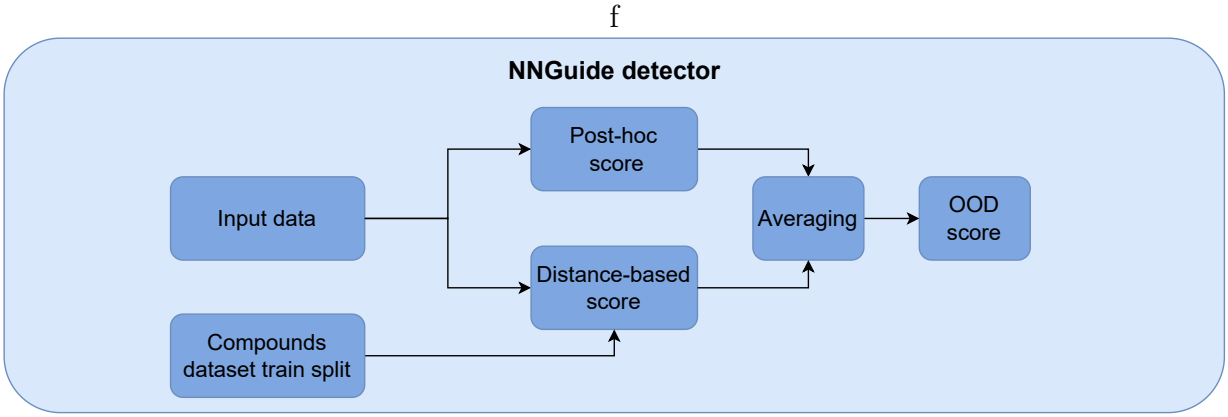


Figure 4.8: Out-Of-Distribution detection score generation overview for NNGuide [59]. The final score is generated for each input by averaging a post-hoc score with a distance-based score.

NNGuide score is a product of the energy score ('base score'), and the KNN score ('guidance' term). The detector scales the Compounds training split data points $\mathbf{x}^{train}$ by the Energy scores generated from logits obtained from the Compounds training split $f(\mathbf{x}^{train})$:

$$\begin{aligned} \text{Energy}(\mathbf{x}, T) &= T \cdot \log \sum_{i=0}^{69} \exp(\frac{f(\mathbf{x}_i)}{T}) \\ \mathbf{x}_{trainScaled} &= \mathbf{x}^{train} \cdot \text{Energy}(\mathbf{x}^{train}, T) \end{aligned} \tag{24}$$

For a test-time input $\mathbf{x}$ we calculate the distance score and the post-hoc score. The distance score is a changed KNN score as defined in 22. The post-hoc score is define as the Energy function (24):

$$\begin{aligned} \text{PostHocScore}\,(\mathbf{x}, T) &= \text{Energy}(\mathbf{x}, T) \\ \text{DistanceScore}\,(\mathbf{x}, k) &= \frac{1}{k} \sum_{j} \left(\langle \mathbf{x}, \mathbf{x}_{(j)}^{trainScaled} \rangle \right) \\ \text{where } j &= \{1, \ldots, k\} \\ \mathbf{x}_{(j)}^{trainScaled} &\text{ is j-th nearest neighbor to the scaled training data} \end{aligned} \tag{25}$$

For a test-time input $\mathbf{x}$ we calculate NNGuide score by weighting the distance score $\text{DistanceScore}\,(\mathbf{x}, k)$ with the post-hoc score $\text{PostHocScore}\,(\mathbf{x}, T)$:

$$\text{NNGuide}\,(\mathbf{x}, T, k) = \text{PostHocScore}\,(\mathbf{x}, T) \cdot \text{DistanceScore}\,(\mathbf{x}, k) \tag{26}$$

In our experiments, we use the temperature $T = 1$ and $k = 10$ for kNN distance.

4.4 OOD detection

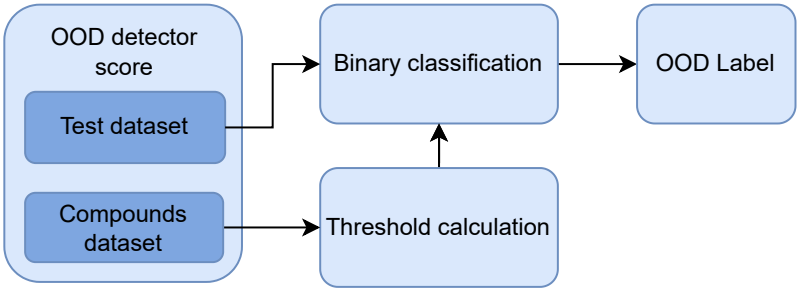


Figure 4.9: OOD binary classification process overview. First, a threshold is obtained from the validation split of the Compounds dataset. The method produces OOD labels for the Test dataset with the threshold as a binary classification.

In this subsection, we explain how the OOD scores (Sec. 4.3) are transformed into a functional OOD detector. The input are the OOD scores obtained from: (i) the validation split of the Compounds dataset, and (ii) the Test dataset. The output are OOD labels for each data point in the Test dataset. Each label indicates whether a data point is In-Distribution or Out-Of-Distribution. See Fig. 4.9 for an overview of this process.

OOD detection uses a **threshold-based binary classification**. Recall the definition of an OOD detector G (from Sec. 4.1). The SVM model f (SVM from Sec. 4.2) is trained on the Compounds dataset. In testing, we deploy f on the Test dataset that is a mixture of ID and OOD. The detector G predicts whether a testing sample $\mathbf{x}$ is In-Distribution (ID) or Out-Of-Distribution (OOD):

$$G(\mathbf{x}, \tau) = \begin{cases} 1 & \text{if } \mathbf{x} \geq \tau \\ 0 & \text{if } \mathbf{x} < \tau \end{cases} \tag{27}$$

Here τ is a threshold derived from the validation split of the Compounds dataset.

Threshold calculation
The input are scores obtained from the validation split of the Compounds dataset. The output is a threshold τ. We use the validation split because we know that the data come from the same distribution on which the SVM model was trained, while the data were not used for the training itself so we reduce a bias.

We set the threshold so that a high fraction of the Compounds dataset is correctly classified as ID. In other words, we change the threshold by setting a True Positive (TP) rate (ID data classified as ID). Alternatively, we can set a False Negative (FN) rate (ID data misclassified as OOD). The implementation of threshold selection is simple. We sort the scores in descending order and determine the threshold so that the top percentage of scores (specified by the TP rate) will be considered as ID.

In our qualitative experiments, we set the TP rate at 95% and 99%. See Sec. 5 for the results. Note that the threshold is different for each OOD detector.

Generating OOD labels with binary classification
The input are scores obtained from a Test dataset and the threshold from the previous step. The output are OOD labels for each input data point.

We classify the Test data based on 27. If a score is greater than or equal to the threshold, it is considered In-Distribution (assigned the value 1). Otherwise, it is considered Out-Of-Distribution (OOD) (assigned the value 0).

5 Experiments

In this section, we describe a series of qualitative and quantitative experiments performed on the datasets described in Sec. 3. In the specific setting of OOD, we had to select a strategy accounting for the fact that only ID labeled data were available to us. For the **quantitative experiments** presented in Sec. 5.1, we artificially created sets of in/out-distribution compounds from the Compounds dataset (Sec. 3.4.2.). *I.e.*, we selected a portion of compounds to be In-Distribution and used them for SVM training and calibration of distance-based methods, and we marked the rest as Out-Of-Distribution. We hypothesized, that the performance of both SVM and OOD methods heavily depends on the in/out-distribution splits of chemical compounds. Therefore, we applied a 10-fold cross-validation for the in/out-distribution sampling, and report performance accordingly.

In the qualitative experiments presented in Sec. 5.2, we investigated the 'Test dataset' – a single GCxGC-TOF-MS measurement introduced in Sec. 3.4.3. We applied the methods reported in the quantitative experiments and we mainly interpret the results by eyeballing.

5.1 OOD detection quantitative experiments

In this experiment, we quantitatively compare performance of eight OOD detection methods. For the evaluation, we use the ID dataset introduced in Sec. 3.4.2. To investigate performance of the OOD detectors, we synthesise the OOD samples.

5.1.1 Experiment setup

Artificially created sets of in/out-distribution from the 'Compounds dataset' are used as input. We choose an 'ID-to-OOD ratio' that denotes a ratio of the Compounds dataset considered ID, the rest as OOD. With this ratio, we **randomly** select a subset of compounds to be OOD by 10-fold cross-validation [13]. E.g., for each fold and ID-to-OOD ratio = 0.9, we randomly select 7 (out of 70) compounds and mark them as OOD and mark the rest as ID. With this id/out-distribution sets, we train the SVM model (on the ID subset), compute OOD scores (on both ID/OOD sets) and compute the performance metrics. Finally, we average the metrics over 10 folds. The evaluation pipeline for a single fold is illustrated in Fig. 5.1.

Note, that we only split the ID dataset to training and validation subsets, we do not use a test subset. The reason is that although the validation test is used to find hyperparameters of the SVM classifier for the classifier-based OOD methods, we believe that it may still be considered as a testing dataset for OOD detection. Moreover, in case of distance-based

[13]Note, that we use cross-validation in two different places. First, we use cross-validation in splitting the compounds into in/out-distribution sets – purpose here is to determine how sensitive the detection methods are to the choice of OOD compounds. Second, we use cross-validation at the SVM training level – here the purpose is to get estimates of amount of overfitting of the SVM model on our data.

OOD methods, finding hyperparameters is not required. Therefore, the validation dataset actually is the test dataset.

Also note, that to evaluate the investigated OOD methods, we apply them to a set of samples constructed from the validation subset of ID dataset, and the OOD subset. The number of in/out-distribution samples in the constructed dataset is not equal. We do not attempt to make the subsets balanced due to a limited availability of labeled data. In [83], sensitivity of various performance metrics is investigated for binary classification in unbalanced classification scenarios. In our experiments, as discussed later, we follow their conclusions and report metrics which give an accurate and intuitive interpretation of performance of classifiers on imbalanced datasets.

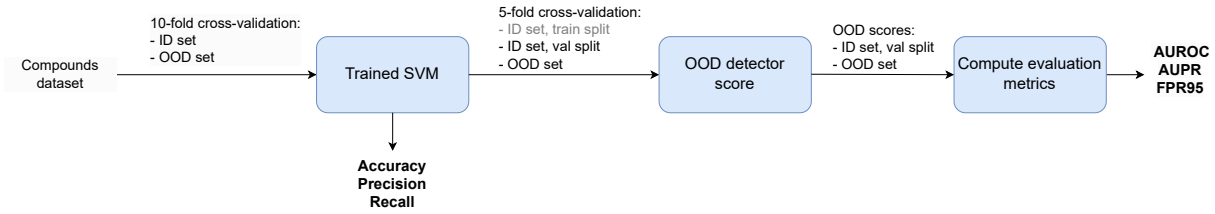


Figure 5.1: Evaluation pipeline for the quantitative experiments. Artificially created sets, using a 10-fold cross-validation, of in/out-distribution compounds from the Compounds dataset are given as input. An SVM model is trained on the ID set using 5-fold cross-validation, and its performance metrics (*i.e.* Accuracy, Precision and Recall) are reported. The validation split of ID set and the OOD set are fed into the SVM to produce OOD detection scores. OOD detection metrics, *i.e.* AUROC, AUPR and FPR95, are calculated from the scores. Outputs of the pipeline is shown in bold.

The output of this pipeline is: (i) ID classification performance metrics and (ii) evaluation metrics for the OOD detection. The parts of this pipeline are explained in Sec. 4.2, 4.4. We conduct experiments for two different values for the ID-to-OOD ratio, specifically 90% and 10%. In Tab. 5.1, we show performance of the SVM: (i) using retention times, (ii) without retention times. We compare the performance of different detection methods in Tab. 5.2. Additionally, we conduct an experiment that uses other than a random sampling for the in/out-distribution sets to evaluate how the selection of compounds we mark OOD affects the detection performance. The results are in Tab. 5.3.

5.1.2 OOD detection performance metrics

The metrics definitions use the following terms. TP represent correctly predicted positive instances, True Negative (TN) are correctly predicted negative instances, False Positive (FP) are instances wrongly predicted as positive, and FN are instances wrongly predicted as negative.

OOD detection is a **threshold-based** binary classification. To assess the performance of a OOD detector (Tab. 5.1, 5.2, 5.3), we use the following metrics. The first metric is **Area Under Receiver Operation Characteristic curve (AUROC)**. Receiver Operation Characteristic (ROC) curve shows the trade-off between True Positive Rate (TPR) and False Positive Rate (FPR) across different decision thresholds. The TPR and FPR are derived from the scores, and we use them to create the ROC curve. The baseline for AUROC = 0.5 which represents a random classifier. We choose AUROC instead of metrics like accuracy because it assesses the classifier's performance across all possible threshold values, rather than being dependent on a specific threshold choice. We compute AUROC by considering the ID samples as the positive class.

Area Under Precision-Recall curve (AUPR) is similar to AUROC but instead of TPR and FPR it plots precision against recall across different decision thresholds. Our choice of AUPR performance metric is based on [83], where the metric is claimed to be suitable for binary classification in unbalanced classification scenarios, which is our case. The computation process is the same as the ROC curve but TPR and FPR are replaced by recall and precision measures. The precision-recall curve shows a trade-off between False Positives and False Negatives. In datasets with a significant imbalance between positive and negative classes, which is our case, AUPR is more informative than AUROC. The baseline for AUPR is equal to the fraction of positive classes [83]. We compute AUPR_{In} by considering the ID samples as the positive class. Similarly, we compute AUPR_{Out} by considering the OOD samples as the positive class.

FPR95 refers to the FPR at which the TPR of ID samples is at 95%. We use this metric to simulate a run of a detector on a dataset where we set the threshold TPR at a fixed value, often 95% (see Sec. 4.4). The objective is to minimize FPR95, unlike AUROC, AUPR_{In} and AUPR_{Out}, where the objective is a maximization.

5.1.3 Classification performance metrics

The SVM model solves a multi-class classification task. We report the SVM perfomance, as the ID classification performance. We use three evaluation metrics:

1. Accuracy
2. Average Precision
3. Average Recall

Accuracy is a commonly used metric in the field of OOD detection. It is expressed as:

$$\text{Accuracy} = \frac{\text{Number of Correct Predictions}}{\text{Total Number of Predictions}} \tag{28}$$

We use a precision and a recall because they offer deeper insights into the model's performance and diagnostic power. They give a nuanced evaluation by capturing the trade-off

OOD detection performance

	performance metric	AUROC ↑		FPR95 ↓		$AUPR_{In}$ ↑		$AUPR_{Out}$ ↑	
	retention times available	✓	✗	✓	✗	✓	✗	✓	✗
classifier-based	SVM+MSP	**75.92**	67.91	**70.41**	87.10	**79.88**	74.57	**65.17**	52.31
	SVM+MaxLogit	**75.68**	67.82	**74.15**	87.22	**79.80**	74.54	**64.48**	52.16
	SVM+KL	**73.17**	66.24	**79.70**	87.18	**78.77**	73.80	**60.03**	51.24
	SVM+Energy	**75.47**	67.73	**77.12**	88.88	**79.73**	74.51	**63.72**	51.96
	SVM+NNGuide	**71.52**	65.12	**80.96**	91.05	**78.47**	75.18	**58.40**	48.00
distance-based	Mahalanobis	**73.03**	72.15	**77.33**	78.00	**84.11**	83.11	**60.02**	59.45
	KNN	47.32	**47.86**	**96.19**	96.49	65.14	**65.25**	33.61	**33.86**
	SSD	**70.15**	70.09	**79.29**	79.49	**80.94**	80.68	**57.49**	57.47

ID-classification SVM performance

performance metric	Precision ↑		Accuracy ↑		Recall ↑	
retention times available	✓	✗	✓	✗	✓	✗
	98.94	96.13	**98.81**	96.08	**98.82**	96.09

Table 5.1: OOD detection methods performance on the compounds classification problem. An artificially created dataset of in/out-distribution compounds with ID-to-OOD ratio = 0.9 was used, with and without retention times available. Performance of SVM on the ID classification problem is also reported, as it is informative w.r.t. the performance of classifier-based OOD detectors.

between False Positives and False Negatives. Recall and precision have different definitions for a multi-class classification. We use ***micro-averaged*** **precision and recall**:

$$\text{Average Precision} = \frac{1}{70} \sum_{\text{Class i}=1}^{70} \frac{\text{TP}_{\text{Class i}}}{\text{TP}_{\text{Class i}} + \text{FP}_{\text{Class i}}} \tag{29}$$

$$\text{Average Recall} = \frac{1}{70} \sum_{\text{Class i}=1}^{70} \frac{\text{TP}_{\text{Class i}}}{\text{TP}_{\text{Class i}} + \text{FN}_{\text{Class i}}} \tag{30}$$

We calculate the evaluation metrics as the average across all 5 folds of the cross-validation.

5.1.4 Results

First, we compare the SVM model: (i) without retention times, (ii) with retention times. Our hypothesis is that incorporating retention times as additional features enhances performance for both ID classification and OOD detection. Results in Tab. 5.1 demonstrate that the **performance increases** for ID classification and OOD detection tasks [14],**when**

[14]KNN detector produces the same performance metrics for SVM model: (i) utilizing retention times, (ii) not utilizing retention times.

the SVM model utilizes retention times as additional features.

OOD detection performance

ID-to-OOD ratio	0.9				0.1			
performance metric	AUROC ↑	FPR95 ↓	$AUPR_{In}$ ↑	$AUPR_{Out}$ ↑	AUROC ↑	FPR95 ↓	$AUPR_{In}$ ↑	$AUPR_{Out}$ ↑
SVM+MSP	**75.92**	**70.41**	79.88	**65.17**	83.60	42.28	10.04	99.56
SVM+MaxLogit	75.68	74.15	79.80	64.48	83.61	42.22	10.04	99.56
SVM+KL	73.17	79.70	78.77	60.03	82.84	43.50	9.85	99.53
SVM+Energy	75.47	77.12	79.73	63.72	83.61	42.21	10.04	99.56
SVM+NNGuide	71.52	80.96	78.47	58.40	78.20	51.91	7.63	99.39
Mahalanobis	73.03	77.33	**84.11**	60.02	**93.78**	**19.84**	**40.91**	**99.85**
KNN	47.32	96.19	65.14	33.61	53.18	87.52	3.22	98.17
SSD	70.15	79.29	80.94	57.49	92.97	20.55	33.50	99.83

ID-classification SVM performance

performance metric	Precision ↑	Accuracy ↑	Recall ↑	Precision ↑	Accuracy ↑	Recall ↑
	98.94	98.81	98.82	99.66	99.64	99.65

Table 5.2: Results for 8 OOD detection methods on an artificially created set of in/out distribution compounds with a random sampling and: (i) ID-to-OOD ratio = 0.9, (ii) ID-to-OOD ratio = 0.1. The average performance across 10-fold cross-validation is reported. Results are produced by the SVM model that uses retention times as additional features. Performance of SVM on the ID classification problem is also reported, as it is informative w.r.t. the performance of classifier-based OOD detectors.

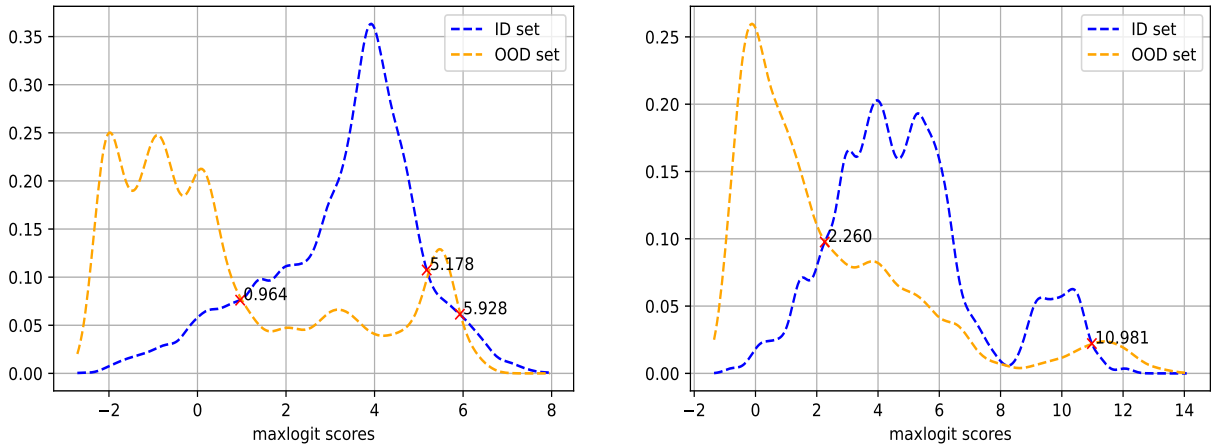
For the next experiment, we want to compare the performance of different detection methods and see how the performance changes for two in/out-distribution sets specified by ID-to-OOD ratios. The reported metrics are averages of 10-fold cross-validation.

Different ratios simulate different scenarios for the detection task, when having a high ratio, *i.e.* 0.9, most compound from a test-time data are In-Distribution. Similarly for a low ratio, *i.e.* 0.1, a minority of compounds are from a test-time data are ID. This resembles the Test dataset used for qualitative experiments (Sec. 3.4.3) as we only have 70 ID compounds and many more OOD compounds. We expect that having SVM trained on a low number of samples, *i.e.* 7, will produce better detection performance as the model should better estimate the distribution of the ID set. We were not surprised by the results presented in Tab. 5.2 which tells that the AUROC and FPR95 values are higher (lower respectively) for ID-to-OOD = 0.1. We also carried out experiments with other ID-to-OOD ratios (0.5, 0.75, 0.95 and 0.98) and identified a consistent pattern that by incorporating more OOD samples results in higher AUROC and lower FPR95 values [15]. In Tab. 5.2 we also see that for ID-to-OOD = 0.1 we get $AUPR_{Out} \approx 99$ for all methods, where the baseline[16] has value $AUPR_{Out} = 90$, which is considered a favorable outcome. Then, $AUPR_{In} > 30$ for the Mahalanobis and SSD detectors and only $AUPR_{In} \approx 10$ for the rest, where the baseline has value $AUPR_{In} = 10$. For ID-to-OOD = 0.9, the methods give satisfactory values of $AUPR_{Out}$ but $AUPR_{In}$ below a baseline, which we consider a

[15] In order not to overwhelm a reader, we decided not to include the results for other ID-to-OOD ratios in the thesis.

[16] The baseline for AUPR is the proportion of positive samples.

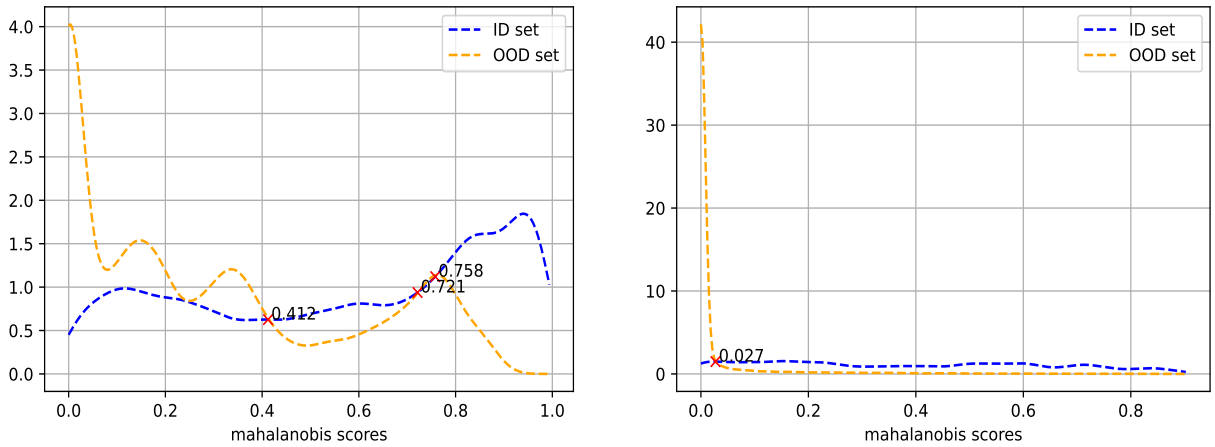
failure. Based on that, we conclude that: (i) distance-based methods, **Mahalanobis and SSD, achieve the best performance**, (ii) **post-hoc methods**, that utilize SVM outputs, **have a lower performance compared to distance-based methods**, and (iii) **OOD detection methods produce satisfactory results** on an artificially in/out distribution set created **with the** ID-to-OOD = 0.1 **ratio and fail on a set created with the** ID-to-OOD = 0.9 **ratio**.



(a) ID-to-OOD ratio = 0.9

(b) ID-to-OOD ratio = 0.1

Figure 5.2: MaxLogit detector OOD detection scores distribution. The scores were computed on an artificially created set of in/out distribution compounds with a random sampling and, left: ID-to-OOD ratio = 0.9, right: ID-to-OOD ratio = 0.1. It corresponds to a fold in a cross-validation. OOD scores produced with retention times available. The red colored crosses indicate where the distributions intersect, an ideal threshold score value.



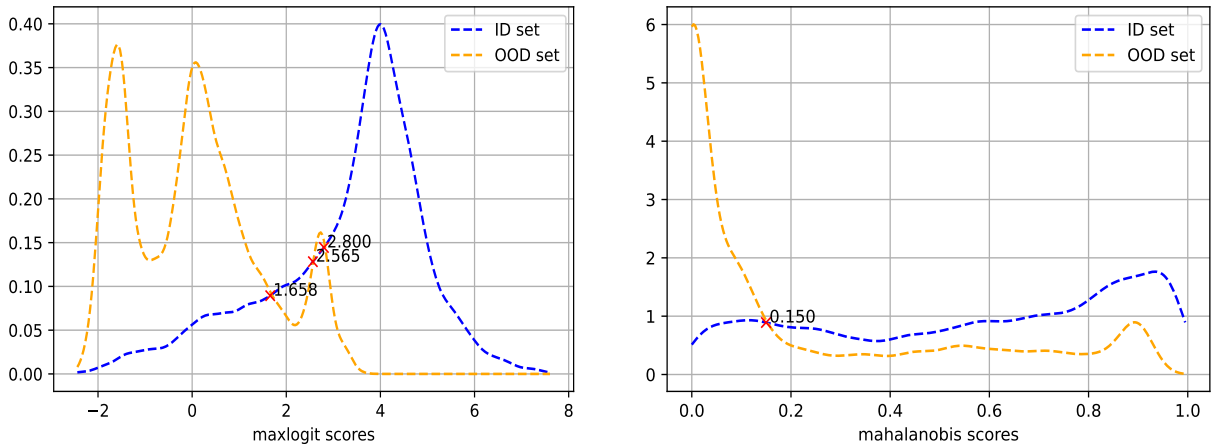
(a) ID-to-OOD ratio = 0.9

(b) ID-to-OOD ratio = 0.1

Figure 5.3: Mahalanobis detector OOD detection scores distribution. The scores were computed on an artificially created set of in/out distribution compounds with a random sampling and, left: ID-to-OOD ratio = 0.9, right: ID-to-OOD ratio = 0.1. It corresponds to a fold in a cross-validation. OOD scores produced with retention times available. The red colored crosses indicate where the distributions intersect, an ideal threshold score value.

We hypothesize, that the reason for the failure of the detection methods is the inability to distinguish small nuances between some of the spectrums, which stem from the quality of the spectra and possible the lack of samples. Next, we examine the OOD scores produced by the methods to further investigate the problem.

An ideal OOD score probability distribution plot for: (i) ID compounds, *i.e.* blue line, has a sharp peak on the right indicating that most of the ID compounds are assigned with large scores, and (ii) OOD compounds, *i.e.* orange line, has a sharp peak on the left indicating most of OOD compounds are assigned with small scores. In Fig. 5.2, Fig. 5.3, we observe two problems: (i) some ID compounds have too low scores, and (ii) some OOD compounds have too high scores.



(a) AUROC = 91.48, FPR95 = 55.30, $\text{AUPR}_{\text{In}} = 95.67$, $\text{AUPR}_{\text{Out}} = 82.53$

(b) AUROC = 80.76, FPR95 = 53.29, $\text{AUPR}_{\text{In}} = 85.70$, $\text{AUPR}_{\text{Out}} = 78.16$

Figure 5.4: **MaxLogit** and **Mahalanobis** detector OOD detection scores distribution. The scores were computed on an artificially created set of in/out distribution compounds with random sampling and, left: ID-to-OOD ratio = 0.9, right: ID-to-OOD ratio = 0.1. It corresponds to a fold in a cross-validation that gives returns the best results. Their performance is denoted under the figures. OOD scores produced with retention times available. The red colored crosses indicate where the distributions intersect, an ideal threshold score value.

We suspect that the results are largely impacted by the choice of selected OOD compounds, which we perform at random. We observe that OOD compounds, that receive high scores, belong to compounds semantically very similar to what we consider ID compounds. *I.e.*, consider two compounds: '9-Hexadecenoic acid, octadecyl ester' and '9-Hexadecenoic acid, hexadecyl ester'. They differ primarily in the length of the alkyl chains attached to the '9-Hexadecenoic acid' molecule when forming the respective esters. Consequently, when we assign an ester of the acid in both id/out-distribution sets, we get high scores for OOD compounds. In Fig. 5.4, there are the results for the MaxLogit detector, on a validation set with different choice of OOD compounds, and the detector does not produce such high scores for OOD compounds, as in Fig. 5.2. As a result, the overall performance is better. We see the same trend for the Mahalanobis detector. Next, we perform a new experiment where we select 5 esters of the '9-Hexadecenoic acid' as OOD compounds. This experiment uses a fixed in/out-distribution sets so we do not use cross-validation.

In Tab. 5.3, there are results for the new experiment. For post-hoc methods, *i.e.* MSP, MaxLogit, KL and Energy, and for ID-to-OOD ratio = 0.93, the performance is slightly better compared to the cross-validation averaged performance but it does not exceed the performance of the best fold. More importantly, the AUPR_{In} and AUPR_{Out} is above the

OOD detection performance

ID-to-OOD ratio	0.93				0.07			
performance metric	AUROC ↑	FPR95 ↓	$AUPR_{In}$ ↑	$AUPR_{Out}$ ↑	AUROC ↑	FPR95 ↓	$AUPR_{In}$ ↑	$AUPR_{Out}$ ↑
SVM+MSP	**84.95**	65.03	**94.68**	**57.94**	19.97	88.34	0.96	97.51
SVM+MaxLogit	84.84	**64.52**	94.65	57.65	19.96	88.32	0.96	97.51
SVM+KL	81.28	68.20	93.04	52.48	18.80	90.19	0.95	97.41
SVM+Energy	84.77	**64.52**	94.63	57.04	19.96	88.32	0.96	97.51
NNGuide	82.04	76.81	93.74	51.99	51.39	84.49	1.48	98.81
Mahalanobis	71.96	71.10	89.66	53.35	**99.44**	**2.91**	**83.92**	**99.99**
KNN	59.41	93.14	84.27	29.35	95.64	32.28	37.31	99.92
SSD	69.76	71.24	87.77	52.33	99.38	3.18	83.16	**99.99**

ID-classification SVM performance

performance metric	Precision ↑	Accuracy ↑	Recall ↑	Precision ↑	Accuracy ↑	Recall ↑
	98.71	98.68	98.69	99.71	99.71	99.71

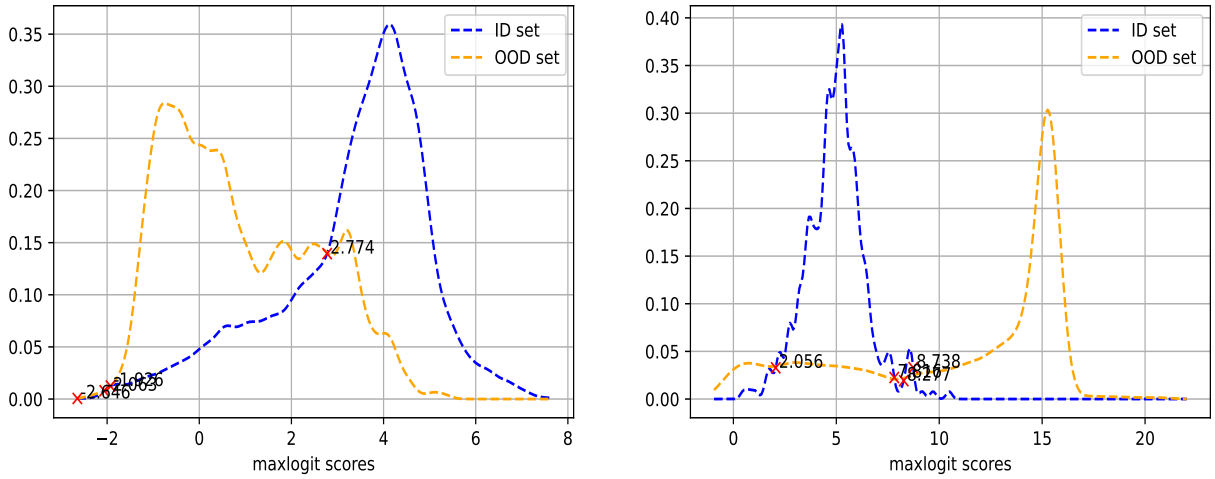
Table 5.3: Results for 8 OOD detection methods on an artificially created set of in/out distribution compounds with compound structure-aware sampling (see 5.1.4 for explanation) and: (i) ID-to-OOD ratio = 0.93, (ii) ID-to-OOD ratio = 0.07. Results are produced by the SVM model that uses retention times as additional features. Performance of SVM on the ID classification problem is also included, as it is informative w.r.t. the performance of classifier-based OOD detectors.

baseline [17], although very slightly, for MSP, MaxLogit, KL, Energy and NNGuide detectors. In Fig. 5.5, we see the comparison of the MaxLogit detector scores produced: (i) with random sampling, and (ii) with selecting only esters of a specific acid. The score distribution for ID compounds is similar but the score distribution for OOD compounds lacks the high scores, *i.e.* around MaxLogit score ≈ 6, for the new experiment setup. Thus, we confirm, the **choice of OOD compounds has a large impact on OOD detection performance**. We still see quite high scores, *i.e.* around MaxLogit score ≈ 3, we assume this is caused by semantical similarities of the acid's esters with other compound. And our SVM model is not able to distinguish similar compounds given the quality of the Compounds dataset. From Tab. 5.3 and Fig. 5.5, we observe that for ID-to-OOD ratio = 0.93, the post-hoc methods fail, producing too high scores for OOD compounds. We believe the reason for the failure is overfitting the ID set containing only 5 compounds.

In Tab. 5.3, we see that performance for distance-based methods, *i.e.* Mahalanobis, SSD, in the new experiment, is similar to the experiment with random sampling, when using ID-to-OOD ratio = 0.93. However, when using ID-to-OOD ratio = 0.07 these two methods achieve AUROC $\approx$ 99, FPR95 $\approx$ 3, $AUPR_{In} \approx 83$ and $AUPR_{Out} \approx 100$ [18]. We conclude that **distance-based methods** are better suited for OOD detection on test datasets including a lot more compounds than the ID classifier has trained with, and they **perfectly separate the id/ood distribution with a structure-aware sampling**, *i.e.* the compound are semantically very similar.

[17] The baseline for $AUPR_{In} = 93$ and the baseline for $AUPR_{Out} = 7$.

[18] The AUROC and FPR95 are not a good performance estimator in this case, due to the class imbalances in the id/ood sets. We still report them as it is a standard in the OOD detection literature.



(a) ID-to-OOD ratio = 0.93

(b) ID-to-OOD ratio = 0.07

Figure 5.5: **MaxLogit** detector OOD detection scores distribution. The scores were computed on an artificially created set of in/out distribution compounds with compound structure-aware sampling (see 5.1.4 for explanation) and, left: (ID-to-OOD ratio = 0.93), right: ID ID-to-OOD ratio = 0.07. OOD scores produced with retention times available. The red colored crosses indicate where the distributions intersect, an ideal threshold score value.

5.2 OOD detection qualitative experiments

For evaluation, we use the Test dataset that is described in Sec. 3.4.3. This dataset represents one measurement of a GCxGC-TOF-MS system and it contains a mixture of ID and OOD samples. There are 715,600 data samples in the dataset. Manually checking the system's output for each of the 715,600 data samples is a tedious and prone-to-failures process. Therefore, we do not have labels for it. In Fig. 5.9 and Fig. 5.10, there are the visual results of gas chromatograph detector responses after OOD detection. We give our interpretation of the results in this subsection.

5.2.1 Experimental setup

The Compounds dataset is used to train the SVM model. In the experiments, we produce two visual results: (i) color coded predictions of gas chromatograph detector response **without OOD detection**, and (ii) color coded predictions of gas chromatograph detector response **after OOD detection**. The evaluation pipeline is visualized in Fig. 5.6 The pipeline steps were explained in Sec. 4.2, Sec. 4.3 and Sec. 4.4. Distance-based methods, *i.e.* Mahalanobis, SSD, produce similar results, so we show results only for the Mahalanobis detector. Other methods, *i.e.* MSP, MaxLogit, KL, Energy, KNN and NNGuide, also produce similar results, and we show results only for the MSP detector.

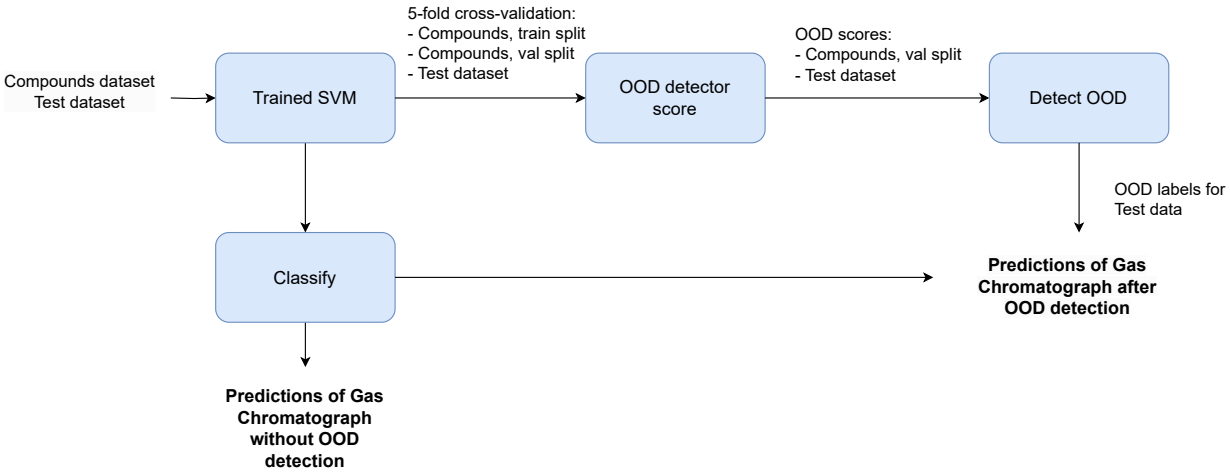


Figure 5.6: Evaluation pipeline for the qualitative experiments. Compounds and Test datasets are given as input. An SVM model is trained on the ID set using 5-fold cross-validation. It is used to classify the Test dataset and produce chromatogram without OOD detection. The validation split of the Compounds dataset and the Test dataset are fed into the SVM to produce OOD detection scores. Chromatogram with OOD detection is produced with the SVM's predictions predicted by the detection. Outputs of the pipeline in bold.

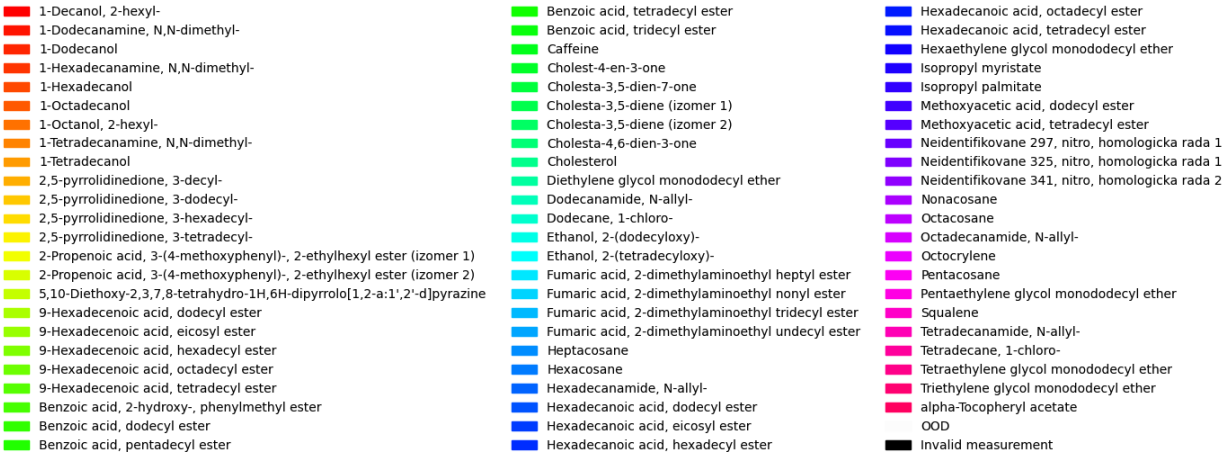


Figure 5.7: Color coding of chemical compounds used in experiments.

We produce **predictions of Gas Chromatograph without OOD detection** by the SVM, trained using 5-fold cross-validation on the Compounds dataset, and classifying the Test dataset, where we save the predicted compounds labels [19]. In order to visualize the compounds as an image we reshape the predictions to a 2D object that represents 460 different retention times $t1$ on y-axis and $2,000$ different retention times t_2 on x-axis. Each point corresponds to the predicted compound label, denoted as a number. We flip the 2D predictions object along y-axis as this visualization is more commonly used. In Fig. 5.8, a result for the predictions without OOD detection is shown.

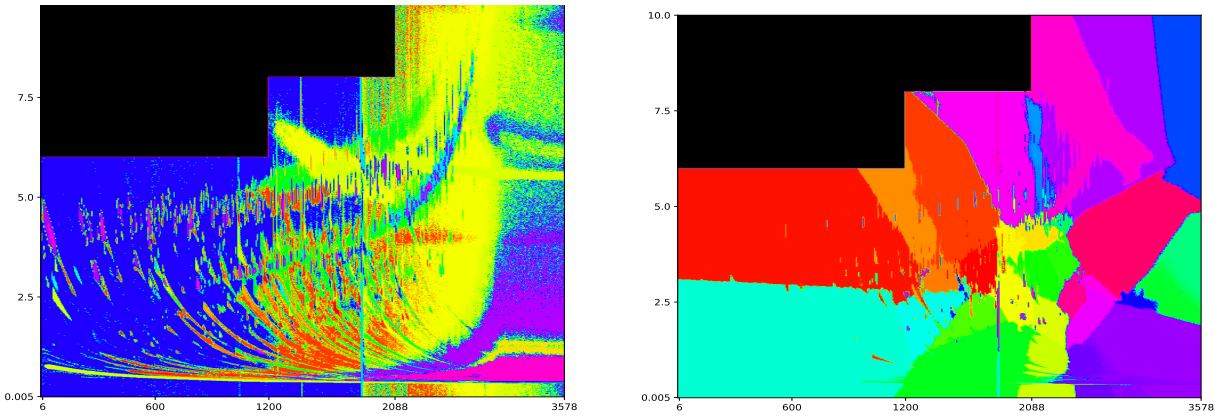
We use a custom colormap (Fig. 5.7) that maps numbered labels for compounds to distinctive colors. Image values correspond to:

- image values 0-69: number of predicted compound (70 total)
- image value 70: Out-Of-Distribution compound
- image value 71: empty measurement

We produce **predictions of Gas Chromatograph after OOD detection** using the same processing. But first, we detect OOD compounds, from Test and Compounds (validation split) datasets OOD detection scores, and mark the associated predicted labels as OOD compounds (depicted as a white color in the image). In Fig. 5.9, Fig. 5.10, results for the predictions with OOD detection are shown.

[19] There are $715,600$ predicted compounds in total.

5.2.2 Results



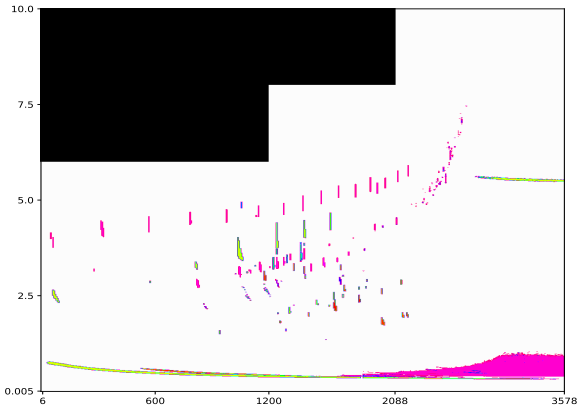
(a) w/o retention times (see Fig. A.1)

(b) w/ retention times (see Fig. A.2)

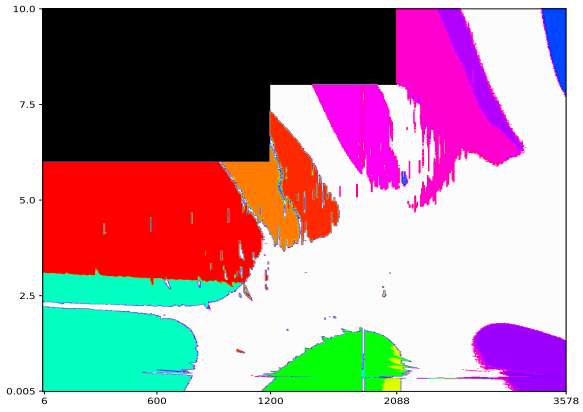
Figure 5.8: Test sample, color coded predictions of **SVM classifier** from gas chromatograph detector responses plotted as a function of the first (x axis), and the second column (y axis) retention time. Left: without, right: with retention times available to SVM. For enlarged see Fig. A.1- A.2

.

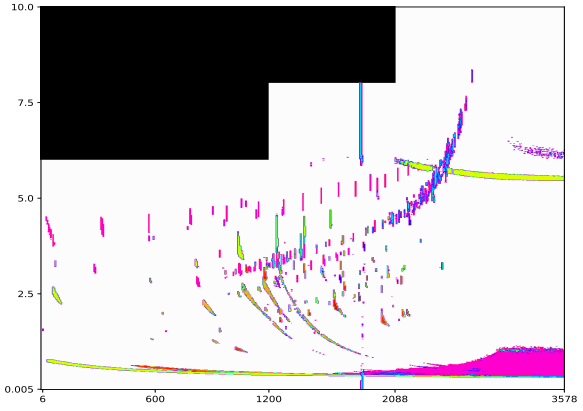
From the character of GCxGC-TOF-MS we expect to see clusters of predicted compounds surrounded by OOD compounds, where the majority are OOD compounds. In Fig. 5.7, is the colormap used in all following visualizations. In Fig. 5.8, you find a comparison with predictions of Gas Chromatograph detector responses, where the prediction were made by an SVM model: (i) without retention times and (ii) with them. The model without retention times produces smaller clusters of classified compounds while the model with retention times produce large clusters that are more separated. We hypothesise, that retention times are valuable features for the SVM model that lead to a better classification of compounds. From the results presented in Fig. 5.8, we find a less 'noisy' chromatogram for SVM's predictions using the retention times. In the visualizations, it is also shown that some predicted compounds are identical for both models. However, there is a great part of the images that differ. It is difficult to draw any conclusion and reject or accept the hypothesis without ground-truth, compound labels, that we do not have.



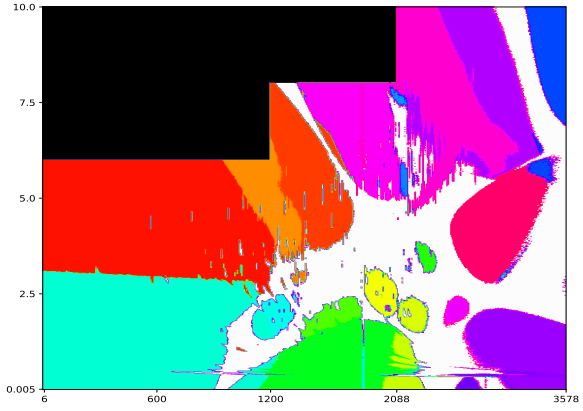
(a) ≈ 1200 OOD training samples, w/o retention times A.3



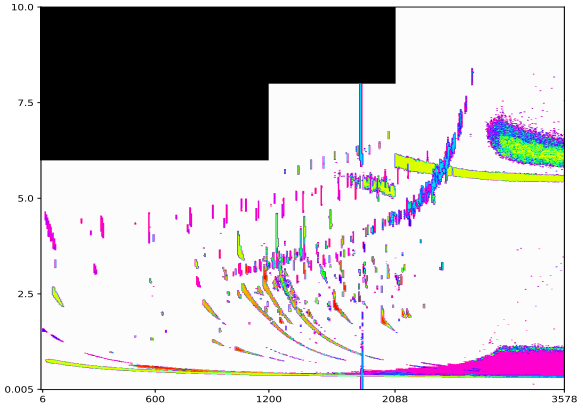
(b) ≈ 1200 OOD training samples, w/ retention times A.4



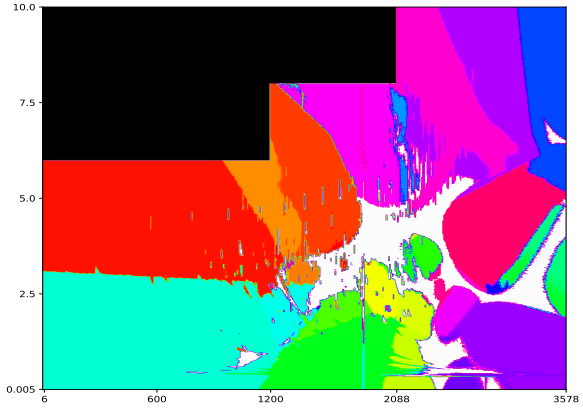
(c) ≈ 480 OOD training samples, w/o retention times A.5



(d) ≈ 480 OOD training samples, w/ retention times A.6



(e) ≈ 240 OOD training samples, w/o retention times A.7

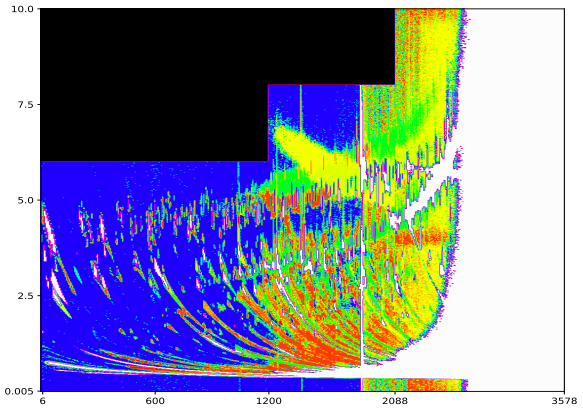


(f) ≈ 240 OOD training samples, w/ retention times A.8

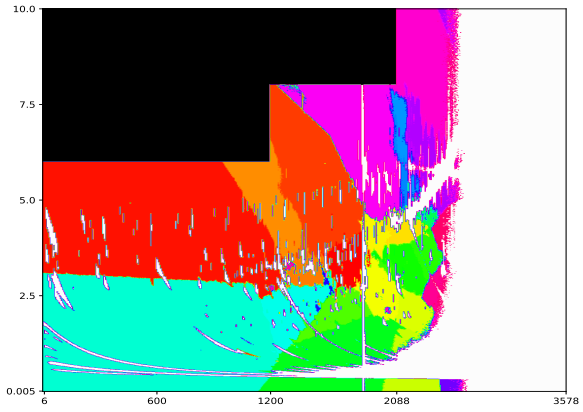
Figure 5.9: Test sample, color coded predictions of SVM classifier with **MSP OOD method** from gas chromatograph detector responses plotted as a function of the first (x axis), and the second column (y axis) retention time. Left: without, right: with retention times available to SVM. Top to bottom: decreasing number of ID samples classified as OOD samples by a threshold-based binary OOD classifier. For enlarged see Fig. A.3–A.8

.

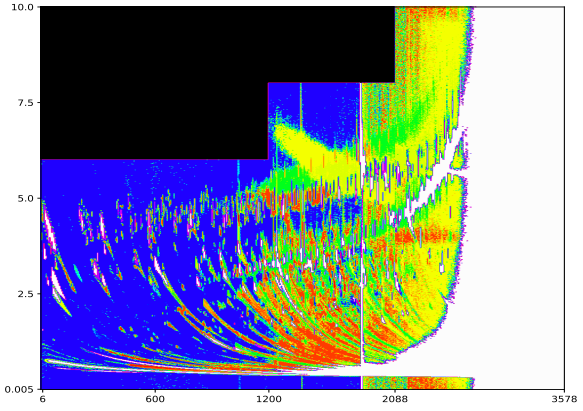
In Fig. 5.9, there are predictions of gas chromatograph detector responses after OOD detection, by the MSP method, in which some classification predictions are rejected by the MSP detector. Which is a post-hoc detection method. We observe that when we utilize retention times as additional feature for the SVM, we classify more compounds as OOD. In order to classify compound as in/out-of-distribution, we need to set a threshold. We assume the provided Compounds dataset, used for training, is not perfect, so we set the threshold to classify some of the compounds from this dataset as OOD samples at a level determined by the number of ID samples classified as OOD. The number controls how much OOD samples is detected. For the model without retention times, we observe that with a lower number of ID samples classified as OOD, generates clusters we are looking for. We also observe that when we descrease this number, we get bigger clusters, that are probably a noise that should be classified as OOD. For the SVM with retention times we observe, that we classify much less compound as OOD, compared to the other SVM model. We hypothetize, that the model predicts some compounds as one of ID compounds, based on the retention times alone, even they are OOD. Inside large clusters of compounds (filled regions), there are smaller clusters of compounds inside them, *i.e.* vertical stripes in the red region, which is something we would like to see. We believe the model can classify some samples well, while it has problems to classify others, and making the prediction based on retention times. A classification model that has been trained on a larger dataset is more likely to produce more accurate outcomes. Since we do not know the exact portion of OOD samples, it is difficult to gain more insights.



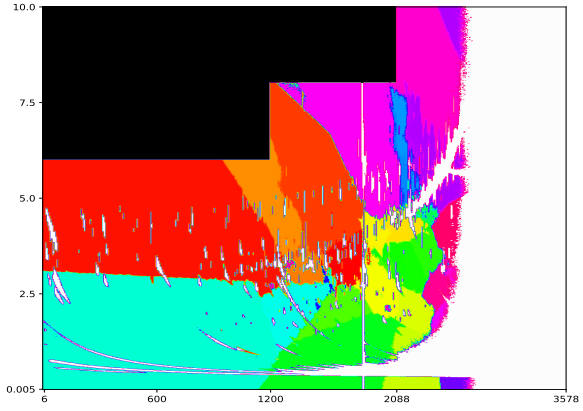
(a) ≈ 1200 OOD training samples, w/o retention times A.9



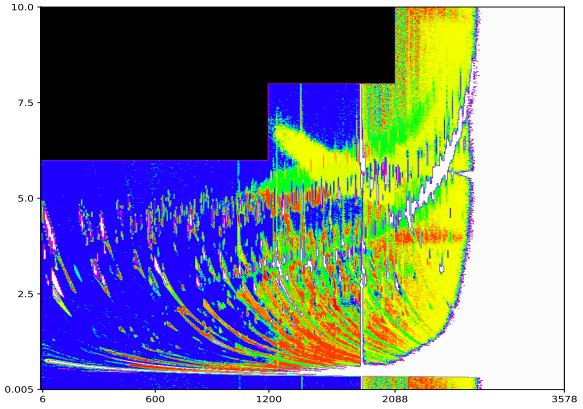
(b) ≈ 1200 OOD training samples, w/ retention times A.10



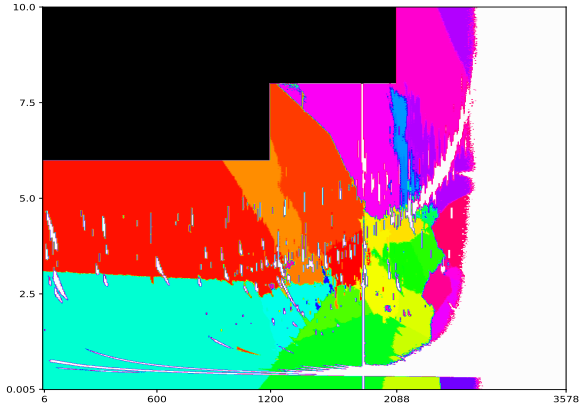
(c) ≈ 480 OOD training samples, w/o retention times A.11



(d) ≈ 480 OOD training samples, w/ retention times A.12



(e) ≈ 240 OOD training samples, w/o retention times A.13

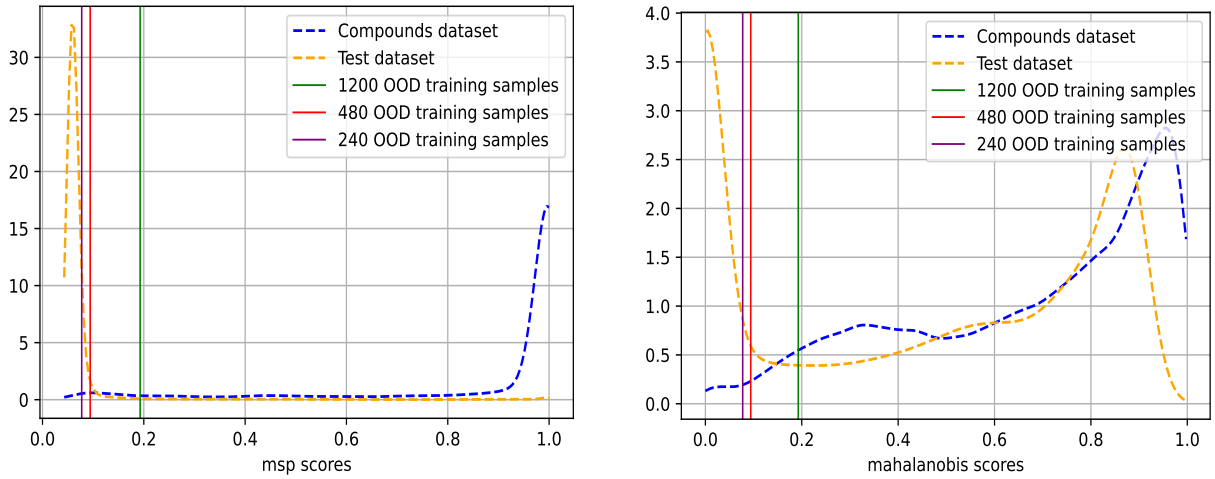


(f) ≈ 240 OOD training samples, w/ retention times A.14

Figure 5.10: Test sample, color coded predictions of **Mahalanobis OOD method** from gas chromatograph detector responses plotted as a function of the first (x axis), and the second column (y axis) retention time. Left: without, right: with retention times available to SVM. Top to bottom: decreasing number of ID samples classified as OOD samples by a threshold-based binary OOD classifier. For enlarged see Fig. A.9– A.14

.

In Fig. 5.10, similar predictions are visualized, produced by the Mahalanobis detector. This is an instance of distance-based detectors, and we observe that the results differ significantly from the MSP detector's predictions. Distance-based methods do not use the SVM model to detect OOD compounds, thus it is not surprising that the compounds predicted as OOD (color coded as white) are identical for SVM with/without retention times. Based on the images, distance-based methods do not work well on the 'Test' dataset, producing OOD labels only for large retention times, thus later in the measurement. We believe this should not be the case for the Test dataset measurement.



(a) MSP detector

(b) Mahalanobis detector

Figure 5.11: **MSP** and **Mahalanobis** detector OOD detection scores distribution. The scores were computed on samples from Test and Compounds datasets. Left: post-hoc MSP, right: distance-based Mahalanobis detector. OOD scores produced without retention times available. Thresholds found with different number of ID samples classified as OOD samples: 240, 480, and 1200 are denoted as vertical lines with different colors.

In Fig. 5.11 OOD detection scores distributions, produced by the MSP and Mahalanobis detector, are presented. We expect to see a sharp peak on the right for the Compounds dataset scores (blue line), which is true for both detectors. As we do not have information about the exact probability distribution of the Test dataset, we can only assume there is a large peak on the left side of the Test dataset (orange line), as well as some adjacent peaks near the large peak of the Compounds dataset scores (blue line). Vertical lines, representing the number of ID samples classified as OOD samples by a threshold-based binary OOD classifier, indicate the binary classifier decision threshold. *I.e.* the Mahalanobis detector predicts all compounds as OOD when they have scores $>= 0.2$, for TPR set at 0.95, which corresponds to classifying $\approx$ 1200 ID samples as OOD. Based on prior knowledge about the Test dataset, we could adjust the threshold to make the ID/OOD separation.

6 Conclusion

In this thesis, we applied 8 OOD detection methods on the chemical compounds classification problem. Since only labels for in-distribution samples were available to us, we artificially created sets of in/out-distribution chemical compounds, and performed a 10-fold cross-validation on the in/out-distribution selection. *i.e.*, we artificially selected various portions of compounds to be in-distribution compounds, and we marked the rest as out-of-distribution.

We performed 2 quantitative experiments on data produced by GCxGC-TOF-MS, an analytic chemistry tool revealing composition of a gas sample. The performance of the OOD methods ranged from AUROC ≈ 76 to AUROC ≈ 99, depending on the ratio of in/out-distribution samples in a test dataset. We specifically evaluated the MSP, MaxLogit, KL and Energy post-hoc OOD methods, all yielding similar performance. The KNN method, a distance-based method, was the only method that did not give satisfactory results.

Generally, performance of the methods heavily depended on the in/out-distribution split, *i.e.* on a particular assignment of compounds to the in/out-distribution sets. This fact is explained by the physico-chemical similarity of particular groups of chemical compounds. For example, there is a group of chemical compounds that only differs in the number of repetitions of a specific chemical element. These compounds appear almost indistinguishable in the output of the GCxGC-TOF-MS. Unsurprisingly, as shown in our experiments, if compounds belonging to such group are scattered between in/out-distribution sets, the performance of OOD detectors is ≈ 10 per cent worse than if they are all assigned to the same in/out-distribution set.

We further performed a series of qualitative experiments on a single test measurement consisting of $\approx 720,000$ samples. We evaluated the results by eyeballing, and we believe that the results support our following conclusion.

Based on our experiments, we believe that OOD detectors are suitable for filtering out chemical compounds not identified as prospective in human scent GCxGC ToF data analysis.

References

[1] Nils Kunze-Szikszay, Maximilian Euler, and Thorsten Perl. Identification of volatile compounds from bacteria by spectrometric methods in medicine diagnostic and other areas: current state and perspectives. *Applied Microbiology and Biotechnology*, 105(16):6245–6255, August 2021.

[2] Anuja Mitra, Sunyoung Choi, Piers R. Boshier, Alexandra Razumovskaya-Hough, Ilaria Belluomo, Patrik Spanel, and George B. Hanna. The Human Skin Volatolome: A Systematic Review of Untargeted Mass Spectrometry Analysis. *Metabolites*, 12(9):824, September 2022.

[3] Fernanda C. O. L. Martins, Michelle A. Sentanin, and Djenaine De Souza. Analytical methods in food additives determination: Compounds with functional applications. *Food Chemistry*, 272:732–750, January 2019.

[4] Petra Pojmanová. Molekulová skladba pachové signatury : disertační práce, 2020.

[5] Yijun Yang, Ruiyuan Gao, and Qiang Xu. Out-of-Distribution Detection with Semantic Mismatch under Masking, July 2022. arXiv:2208.00446 [cs].

[6] Jingkang Yang, Kaiyang Zhou, Yixuan Li, and Ziwei Liu. Generalized Out-of-Distribution Detection: A Survey, August 2022. arXiv:2110.11334 [cs].

[7] Dan Hendrycks and Kevin Gimpel. A baseline for detecting misclassified and out-of-distribution examples in neural networks. October 2016.

[8] Shiyu Liang, Yixuan Li, and R. Srikant. Enhancing the reliability of out-of-distribution image detection in neural networks. *6th International Conference on Learning Representations, ICLR 2018 - Conference Track Proceedings*, June 2017. Publisher: International Conference on Learning Representations, ICLR.

[9] Chuan Guo, Geoff Pleiss, Yu Sun, and Kilian Q. Weinberger. On Calibration of Modern Neural Networks, June 2017.

[10] Weitang Liu, Xiaoyun Wang, John D. Owens, and Yixuan Li. Energy-based out-of-distribution detection. October 2020.

[11] Yann Lecun, Sumit Chopra, and Raia Hadsell. A tutorial on energy-based learning. January 2006.

[12] Chandramouli Shama Sastry and Sageev Oore. Detecting Out-of-Distribution Examples with In-distribution Examples and Gram Matrices, January 2020. arXiv:1912.12510 [cs, stat].

[13] Alex Krizhevsky. Learning Multiple Layers of Features from Tiny Images. *University of Toronto*, May 2012.

[14] Yiyou Sun, Chuan Guo, and Yixuan Li. ReAct: Out-of-distribution Detection With Rectified Activations, November 2021. arXiv:2111.12797 [cs].

[15] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. ImageNet: A large-scale hierarchical image database. In *2009 IEEE Conference on Computer Vision and Pattern Recognition*, pages 248–255, June 2009. ISSN: 1063-6919.

[16] Ziqian Lin, Sreya Dutta Roy, and Yixuan Li. MOOD: Multi-level Out-of-distribution Detection, April 2021. arXiv:2104.14726 [cs].

[17] Dan Hendrycks, Steven Basart, Mantas Mazeika, Andy Zou, Joe Kwon, Mohammadreza Mostajabi, Jacob Steinhardt, and Dawn Song. Scaling Out-of-Distribution Detection for Real-World Settings, May 2022. arXiv:1911.11132 [cs].

[18] Yiyou Sun and Yixuan Li. DICE: Leveraging Sparsification for Out-of-Distribution Detection, July 2022. arXiv:2111.09805 [cs].

[19] Xin Dong, Junfeng Guo, Ang Li, Wei-Te Ting, Cong Liu, and H. T. Kung. Neural Mean Discrepancy for Efficient Out-of-Distribution Detection, March 2022. arXiv:2104.11408 [cs].

[20] Andrija Djurisic, Nebojsa Bozanic, Arjun Ashok, and Rosanne Liu. Extremely Simple Activation Shaping for Out-of-Distribution Detection, May 2023. arXiv:2209.09858 [cs].

[21] Apoorv Vyas, Nataraj Jammalamadaka, Xia Zhu, Dipankar Das, Bharat Kaul, and Theodore L. Willke. Out-of-Distribution Detection Using an Ensemble of Self Supervised Leave-out Classifiers, September 2018. arXiv:1809.03576 [cs, stat].

[22] Matthias Hein, Maksym Andriushchenko, and Julian Bitterwolf. Why ReLU networks yield high-confidence predictions far away from the training data and how to mitigate the problem, May 2019. arXiv:1812.05720 [cs, stat].

[23] Sunil Thulasidasan, Gopinath Chennupati, Jeff Bilmes, Tanmoy Bhattacharya, and Sarah Michalak. On Mixup Training: Improved Calibration and Predictive Uncertainty for Deep Neural Networks, January 2020. arXiv:1905.11001 [cs, stat].

[24] Sangdoo Yun, Dongyoon Han, Seong Joon Oh, Sanghyuk Chun, Junsuk Choe, and Youngjoon Yoo. CutMix: Regularization Strategy to Train Strong Classifiers with Localizable Features, August 2019. arXiv:1905.04899 [cs].

[25] Dan Hendrycks, Andy Zou, Mantas Mazeika, Leonard Tang, Bo Li, Dawn Song, and Jacob Steinhardt. PixMix: Dreamlike Pictures Comprehensively Improve Safety Measures, March 2022. arXiv:2112.05135 [cs].

[26] Yen-Chang Hsu, Yilin Shen, Hongxia Jin, and Zsolt Kira. Generalized ODIN: Detecting out-of-distribution image without learning from out-of-distribution data. February 2020.

[27] Alexander Meinke and Matthias Hein. Towards neural networks that provably know when they don't know, February 2020. arXiv:1909.12180 [cs, stat].

[28] Yuval Netzer, Tao Wang, Adam Coates, Alessandro Bissacco, Bo Wu, and Andrew Y. Ng. Reading Digits in Natural Images with Unsupervised Feature Learning. In *NIPS Workshop on Deep Learning and Unsupervised Feature Learning 2011*, 2011.

[29] David Macêdo, Tsang Ing Ren, Cleber Zanchettin, Adriano L. I. Oliveira, and Teresa Ludermir. Entropic Out-of-Distribution Detection. In *2021 International Joint Conference on Neural Networks (IJCNN)*, pages 1–8, July 2021. arXiv:1908.05569 [cs, stat].

[30] David Macêdo and Teresa Ludermir. Enhanced Isotropy Maximization Loss: Seamless and High-Performance Out-of-Distribution Detection Simply Replacing the SoftMax Loss, April 2022. arXiv:2105.14399 [cs].

[31] Hongxin Wei, Renchunzi Xie, Hao Cheng, Lei Feng, Bo An, and Yixuan Li. Mitigating Neural Network Overconfidence with Logit Normalization, June 2022. arXiv:2205.09310 [cs].

[32] Dan Hendrycks, Mantas Mazeika, and Thomas Dietterich. Deep Anomaly Detection with Outlier Exposure, January 2019. arXiv:1812.04606 [cs, stat].

[33] Yi Li and Nuno Vasconcelos. Background Data Resampling for Outlier-Aware Classification. pages 13218–13227, 2020.

[34] Sina Mohseni, Mandar Pitale, J. B. S. Yadawa, and Zhangyang Wang. Self-Supervised Learning for Generalizable Out-of-Distribution Detection. *Proceedings of the AAAI Conference on Artificial Intelligence*, 34(04):5216–5223, April 2020. Number: 04.

[35] Aristotelis-Angelos Papadopoulos, Mohammad Reza Rajati, Nazim Shaikh, and Jiamian Wang. Outlier Exposure with Confidence Control for Out-of-Distribution Detection. *Neurocomputing*, 441:138–150, June 2021. arXiv:1906.03509 [cs, stat].

[36] Julian Bitterwolf, Alexander Meinke, and Matthias Hein. Certifiably Adversarially Robust Detection of Out-of-Distribution Data, March 2021. arXiv:2007.08473 [cs, stat].

[37] Jingkang Yang, Haoqi Wang, Litong Feng, Xiaopeng Yan, Huabin Zheng, Wayne Zhang, and Ziwei Liu. Semantically Coherent Out-of-Distribution Detection, August 2021.

[38] Jiayu Wu. Tiny ImageNet Challenge. 2017.

[39] Julian Katz-Samuels, Julia Nakhleh, Robert Nowak, and Yixuan Li. Training OOD Detectors in their Natural Habitats, June 2022. arXiv:2202.03299 [cs].

[40] Yifei Ming, Ying Fan, and Yixuan Li. POEM: Out-of-Distribution Detection with Posterior Sampling, June 2022. arXiv:2206.13687 [cs].

[41] Rui Huang, Andrew Geng, and Yixuan Li. On the Importance of Gradients for Detecting Distributional Shifts in the Wild, October 2021. arXiv:2110.00218 [cs].

[42] Richard Dykstra. Kullback–Leibler Information. July 2005.

[43] Yarin Gal and Zoubin Ghahramani. Dropout as a Bayesian Approximation: Representing Model Uncertainty in Deep Learning, October 2016. arXiv:1506.02142 [cs, stat].

[44] Balaji Lakshminarayanan, Alexander Pritzel, and Charles Blundell. Simple and Scalable Predictive Uncertainty Estimation using Deep Ensembles, November 2017. arXiv:1612.01474 [cs, stat].

[45] Andrey Malinin and Mark Gales. Predictive Uncertainty Estimation via Prior Networks, November 2018. arXiv:1802.10501 [cs, stat].

[46] Jay Nandy, Wynne Hsu, and Mong Li Lee. Towards Maximizing the Representation Gap between In-Domain & Out-of-Distribution Examples, January 2021. arXiv:2010.10474 [cs].

[47] Keunseo Kim, JunCheol Shin, and Heeyoung Kim. Locally Most Powerful Bayesian Test for Out-of-Distribution Detection using Deep Generative Models. In *Advances in Neural Information Processing Systems*, volume 34, pages 14913–14924. Curran Associates, Inc., 2021.

[48] Yifei Ming, Ziyang Cai, Jiuxiang Gu, Yiyou Sun, Wei Li, and Yixuan Li. Delving into Out-of-Distribution Detection with Vision-Language Representations, November 2022. arXiv:2211.13445 [cs].

[49] Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, Gretchen Krueger, and Ilya Sutskever. Learning Transferable Visual Models From Natural Language Supervision, February 2021. arXiv:2103.00020 [cs].

[50] Sepideh Esmaeilpour, Bing Liu, Eric Robertson, and Lei Shu. Zero-Shot Out-of-Distribution Detection Based on the Pre-trained Model CLIP. *Proceedings of the AAAI Conference on Artificial Intelligence*, 36(6):6568–6576, June 2022. arXiv:2109.02748 [cs].

[51] Tomas Vojir, Jan Sochman, Rahaf Aljundi, and Jiri Matas. Calibrated Out-of-Distribution Detection with a Generic Representation, September 2023. arXiv:2303.13148 [cs].

[52] Filip Radenovic, Abhimanyu Dubey, Abhishek Kadian, Todor Mihaylov, Simon Vandenhende, Yash Patel, Yi Wen, Vignesh Ramanathan, and Dhruv Mahajan. Filtering, Distillation, and Hard Negatives for Vision-Language Pre-Training, March 2023. arXiv:2301.02280 [cs].

[53] Kimin Lee, Kibok Lee, Honglak Lee, and Jinwoo Shin. A Simple Unified Framework for Detecting Out-of-Distribution Samples and Adversarial Attacks, October 2018. arXiv:1807.03888 [cs, stat].

[54] Jie Ren, Stanislav Fort, Jeremiah Liu, Abhijit Guha Roy, Shreyas Padhy, and Balaji Lakshminarayanan. A Simple Fix to Mahalanobis Distance for Improving Near-OOD Detection, June 2021. arXiv:2106.09022 [cs].

[55] Vikash Sehwag, Mung Chiang, and Prateek Mittal. SSD: A Unified Framework for Self-Supervised Outlier Detection, March 2021. arXiv:2103.12051 [cs].

[56] Yiyou Sun, Yifei Ming, Xiaojin Zhu, and Yixuan Li. Out-of-Distribution Detection with Deep Nearest Neighbors, December 2022. arXiv:2204.06507 [cs].

[57] Yifei Ming, Yiyou Sun, Ousmane Dia, and Yixuan Li. How to Exploit Hyperspherical Embeddings for Out-of-Distribution Detection?, April 2023. arXiv:2203.04450 [cs].

[58] Haoqi Wang, Zhizhong Li, Litong Feng, and Wayne Zhang. ViM: Out-Of-Distribution with Virtual-logit Matching, March 2022. arXiv:2203.10807 [cs].

[59] Jaewoo Park, Yoon Gyo Jung, and Andrew Beng Jin Teoh. Nearest Neighbor Guidance for Out-of-Distribution Detection, September 2023. arXiv:2309.14888 [cs].

[60] Ev Zisselman and Aviv Tamar. Deep Residual Flow for Out of Distribution Detection, July 2020. arXiv:2001.05419 [cs, stat].

[61] Diederik P. Kingma and Prafulla Dhariwal. Glow: Generative Flow with Invertible 1x1 Convolutions, July 2018. arXiv:1807.03039 [cs, stat].

[62] Eric Nalisnick, Akihiro Matsukawa, Yee Whye Teh, Dilan Gorur, and Balaji Lakshminarayanan. Do Deep Generative Models Know What They Don't Know?, February 2019. arXiv:1810.09136 [cs, stat].

[63] Polina Kirichenko, Pavel Izmailov, and Andrew Gordon Wilson. Why Normalizing Flows Fail to Detect Out-of-Distribution Data, June 2020. arXiv:2006.08545 [cs, stat].

[64] Joan Serrà, David Álvarez, Vicenç Gómez, Olga Slizovskaia, José F. Núñez, and Jordi Luque. Input complexity and out-of-distribution detection with likelihood-based generative models, January 2020. arXiv:1909.11480 [cs, stat].

[65] Jingkang Yang, Kaiyang Zhou, and Ziwei Liu. Full-Spectrum Out-of-Distribution Detection, April 2022. arXiv:2204.05306 [cs].

[66] Taylor Denouden, Rick Salay, Krzysztof Czarnecki, Vahdat Abdelzad, Buu Phan, and Sachin Vernekar. Improving Reconstruction Autoencoder Out-of-distribution Detection with Mahalanobis Distance, December 2018. arXiv:1812.02765 [cs, stat].

[67] Olga Vyviurska, Nemanja Koljančić, Ha Anh Thai, Roman Gorovenko, and Ivan Špánik. Classification of Botrytized Wines Based on Producing Technology Using Flow-Modulated Comprehensive Two-Dimensional Gas Chromatography. *Foods*, 10(4):876, April 2021. Number: 4 Publisher: Multidisciplinary Digital Publishing Institute.

[68] Valentijn G. van Mispelaar, Hans-Gerd Janssen, Albert C. Tas, and Peter J. Schoenmakers. Novel system for classifying chromatographic applications, exemplified by comprehensive two-dimensional gas chromatography and multivariate analysis. *Journal of Chromatography. A*, 1071(1-2):229–237, April 2005.

[69] Matteo Croci, Piotr Morawiecki, Valentin Sulzer, and Florian Theil. Classification of Two-Dimensional Gas Chromatography Data, May 2021.

[70] Evan Peter Clark. *Machine Learning Classification of Gas Chromatography Data*. PhD thesis, Virginia Tech, August 2023.

[71] Ian D. Wilson, E. R. Adlard, Michael Cooke, and C. F. Poole. *Encyclopedia of separation science*. Academic Press, San Diego, 2000. OCLC: 44786700.

[72] Harold M. McNair and James M. Miller. *Basic Gas Chromatography*. John Wiley & Sons, 1997.

[73] O.D. Sparkman, Z. Penton, and F.G. Kitson. Gas Chromatography and Mass Spectrometry: A Practical Guide. *Gas Chromatography and Mass Spectrometry: A Practical Guide*, January 2011.

[74] Juliane Elisa Welke and Cláudia Zini. Comprehensive Two-Dimensional Gas Chromatography for Analysis of Volatile Compounds in Foods and Beverages. *Journal of the Brazilian Chemical Society*, 22:609–622, June 2011.

[75] Jens Dallüge, Jan Beens, and Udo A. Th Brinkman. Comprehensive two-dimensional gas chromatography: a powerful and versatile analytical tool. *Journal of Chromatography A*, 1000(1):69–108, June 2003.

[76] Ahmed Mostafa, Matthew Edwards, and Tadeusz Górecki. Optimization aspects of comprehensive two-dimensional gas chromatography. *Journal of Chromatography. A*, 1255:38–55, September 2012.

[77] Philip Marriott and Robert Shellie. Principles and applications of comprehensive two-dimensional gas chromatography. *TrAC Trends in Analytical Chemistry*, 21(9):573–583, September 2002.

[78] Weeraya Khummueng, James Harynuk, and Philip J. Marriott. Modulation ratio in comprehensive two-dimensional gas chromatography. *Analytical Chemistry*, 78(13):4578–4587, July 2006.

[79] Li Deng. The MNIST Database of Handwritten Digit Images for Machine Learning Research [Best of the Web]. *IEEE Signal Processing Magazine*, 29(6):141–142, November 2012. Conference Name: IEEE Signal Processing Magazine.

[80] Chih-Chung Chang and Chih-Jen Lin. LIBSVM: A library for support vector machines. *ACM Transactions on Intelligent Systems and Technology*, 2(3):27:1–27:27, 2011.

[81] Tingyao Wu, Chih-Jen Lin, and R. C. Weng. Probability Estimates for Multi-class Classification by Pairwise Coupling. *J. Mach. Learn. Res.*, December 2003.

[82] John Platt. Probabilistic Outputs for Support Vector Machines and Comparisons to Regularized Likelihood Methods. *Adv. Large Margin Classif.*, 10, June 2000.

[83] Takaya Saito and Marc Rehmsmeier. The Precision-Recall Plot Is More Informative than the ROC Plot When Evaluating Binary Classifiers on Imbalanced Datasets. *PLOS ONE*, 10(3):e0118432, 2015. Publisher: Public Library of Science.

A Qualitative experiments

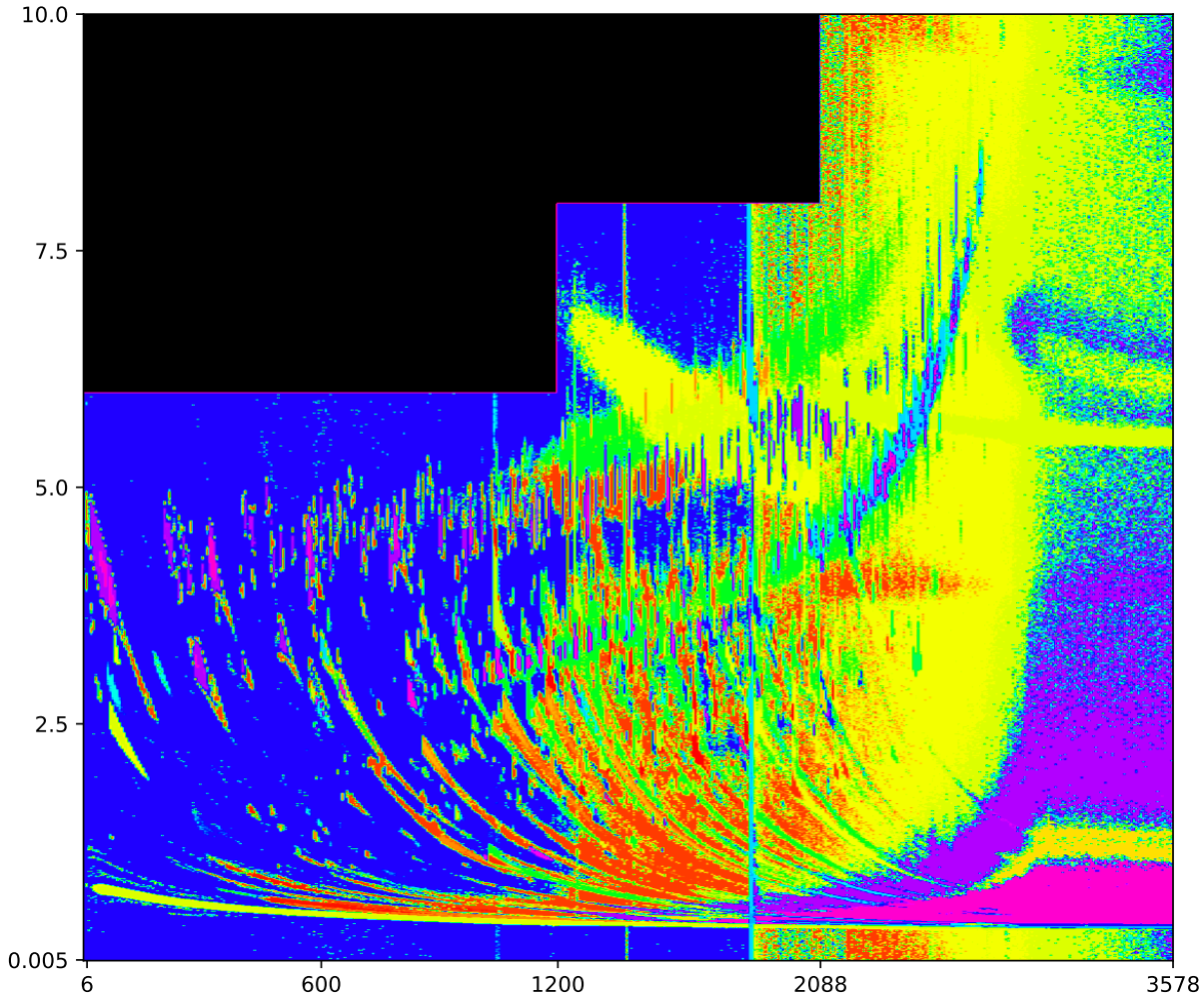


Figure A.1: Test sample, color coded predictions of gas chromatograph detector responses, predictions made without retention times 5.8a

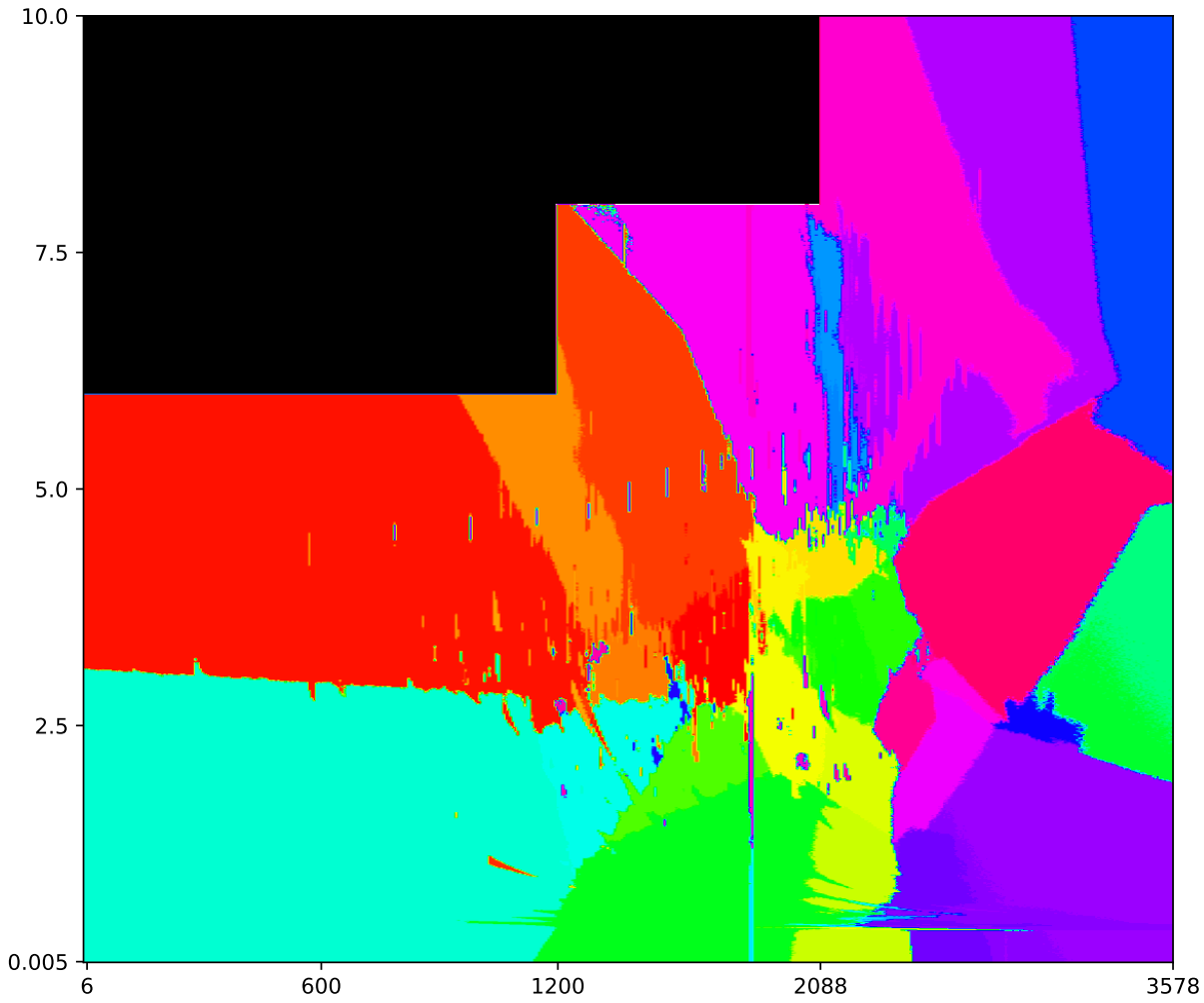


Figure A.2: Test sample, color coded predictions of gas chromatograph detector responses, predictions made with retention times 5.8b

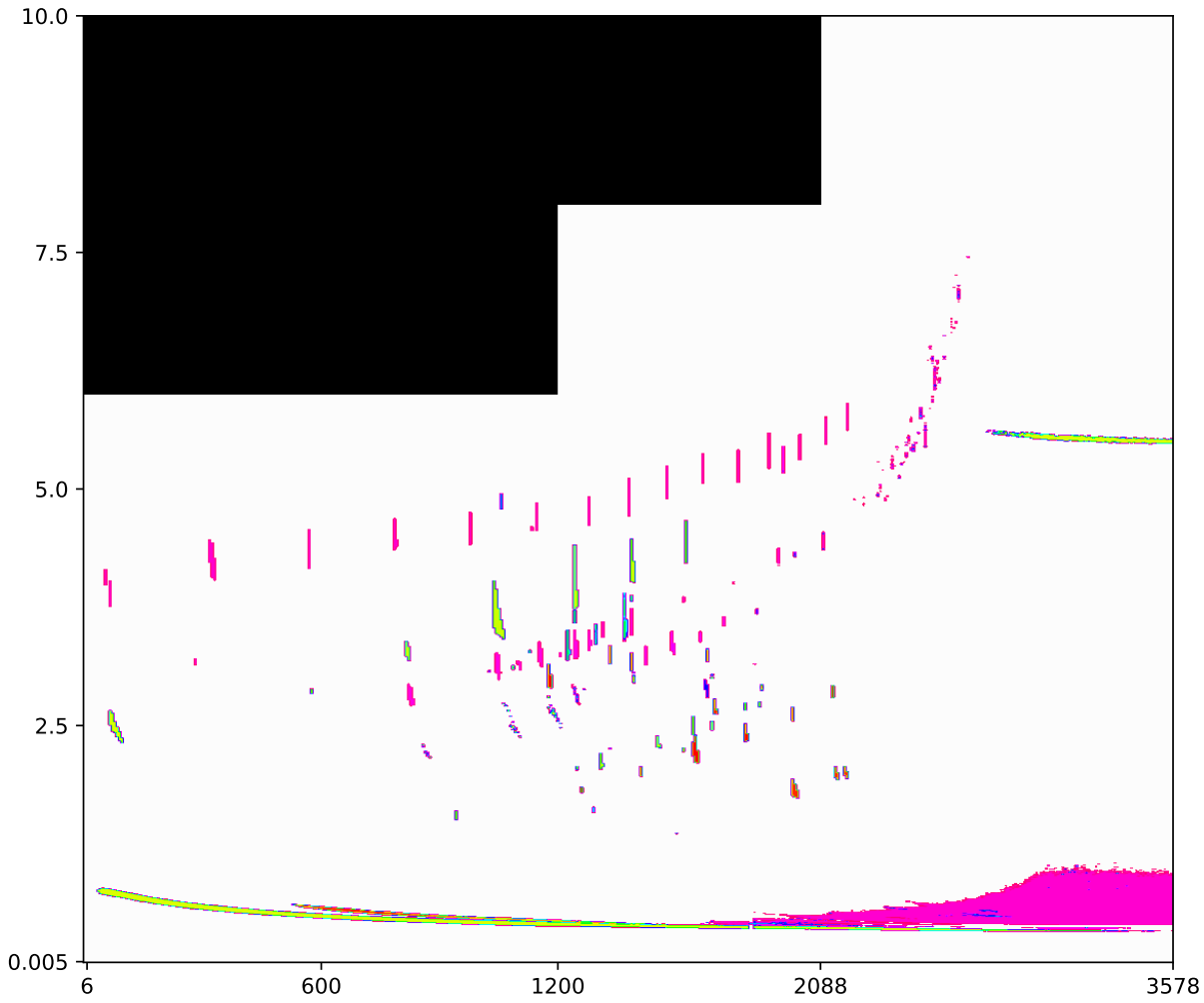


Figure A.3: Test sample, MSP OOD method, color coded predictions of gas chromatograph detector responses, predictions made without retention times, having 1200 ID samples classified as OOD samples by a threshold-based binary OOD classifier 5.9a

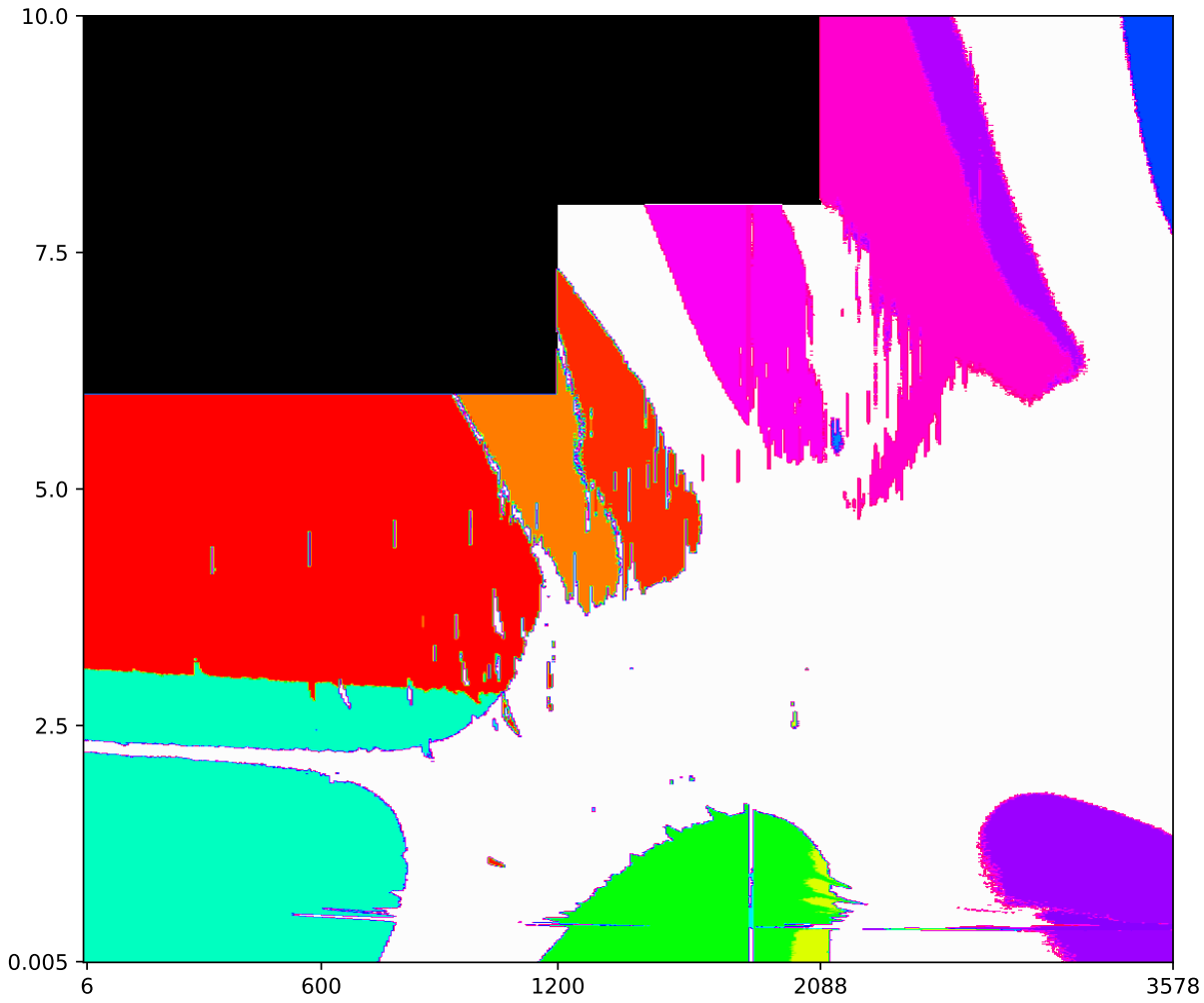


Figure A.4: Test sample, MSP OOD method, color coded predictions of gas chromatograph detector responses, predictions made with retention times, having 1200 ID samples classified as OOD samples by a threshold-based binary OOD classifier 5.9b

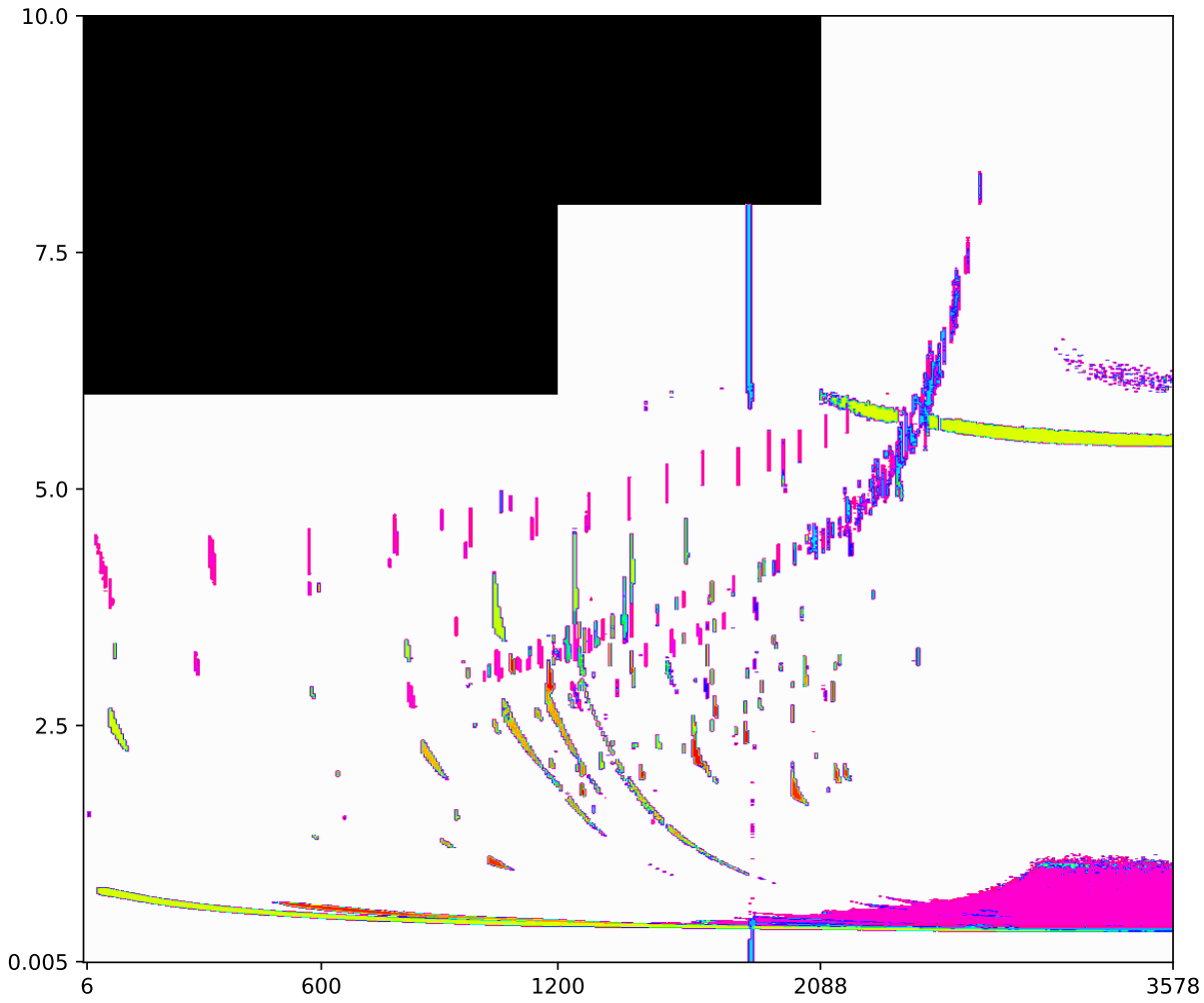


Figure A.5: Test sample, MSP OOD method, color coded predictions of gas chromatograph detector responses, predictions made without retention times, having 460 ID samples classified as OOD samples by a threshold-based binary OOD classifier 5.9c

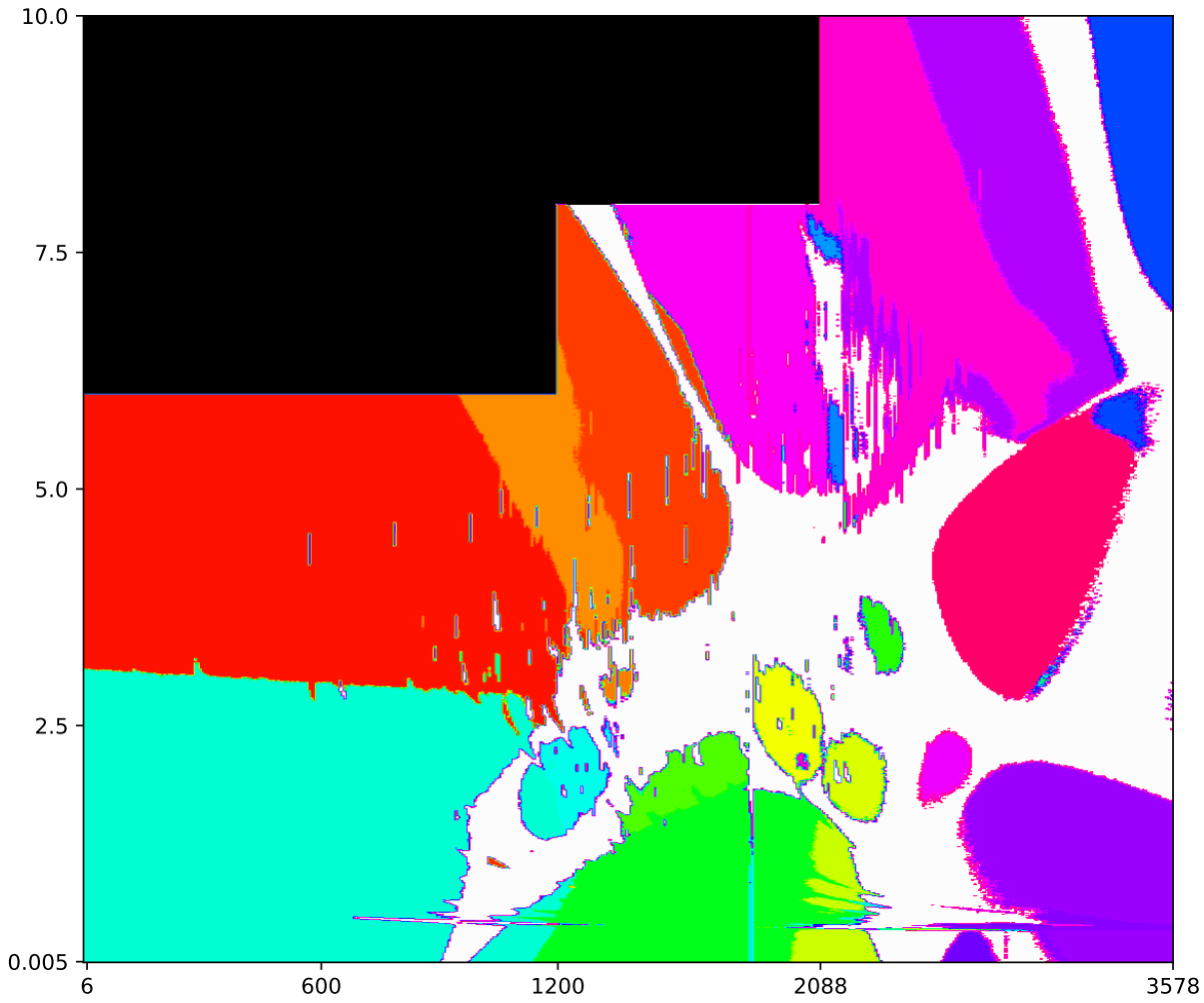


Figure A.6: Test sample, MSP OOD method, color coded predictions of gas chromatograph detector responses, predictions made with retention times, having 460 ID samples classified as OOD samples by a threshold-based binary OOD classifier 5.9d

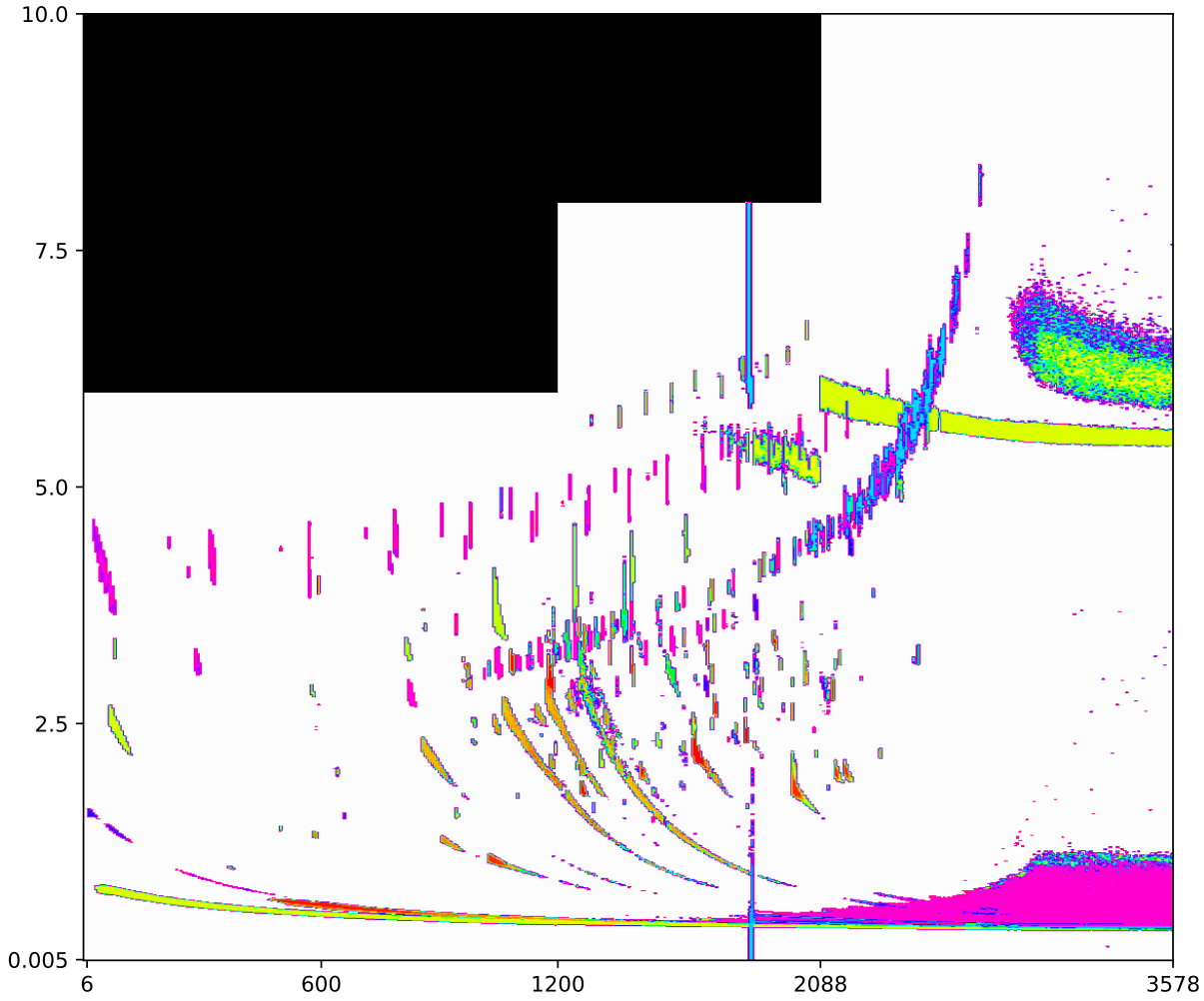


Figure A.7: Test sample, MSP OOD method, color coded predictions of gas chromatograph detector responses, predictions made without retention times, having 240 ID samples classified as OOD samples by a threshold-based binary OOD classifier 5.9e

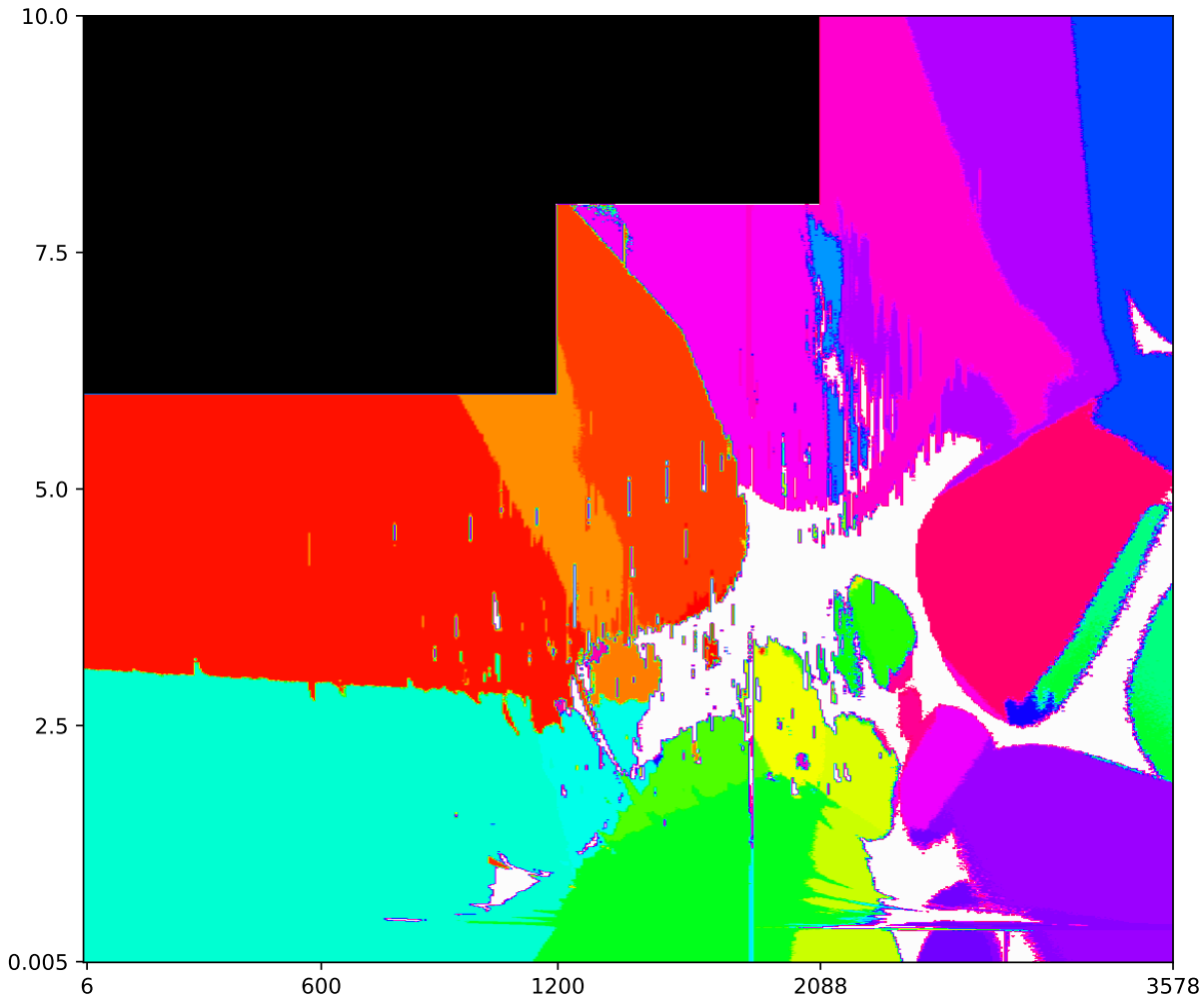


Figure A.8: Test sample, MSP OOD method, color coded predictions of gas chromatograph detector responses, predictions made with retention times, having 240 ID samples classified as OOD samples by a threshold-based binary OOD classifier 5.9f

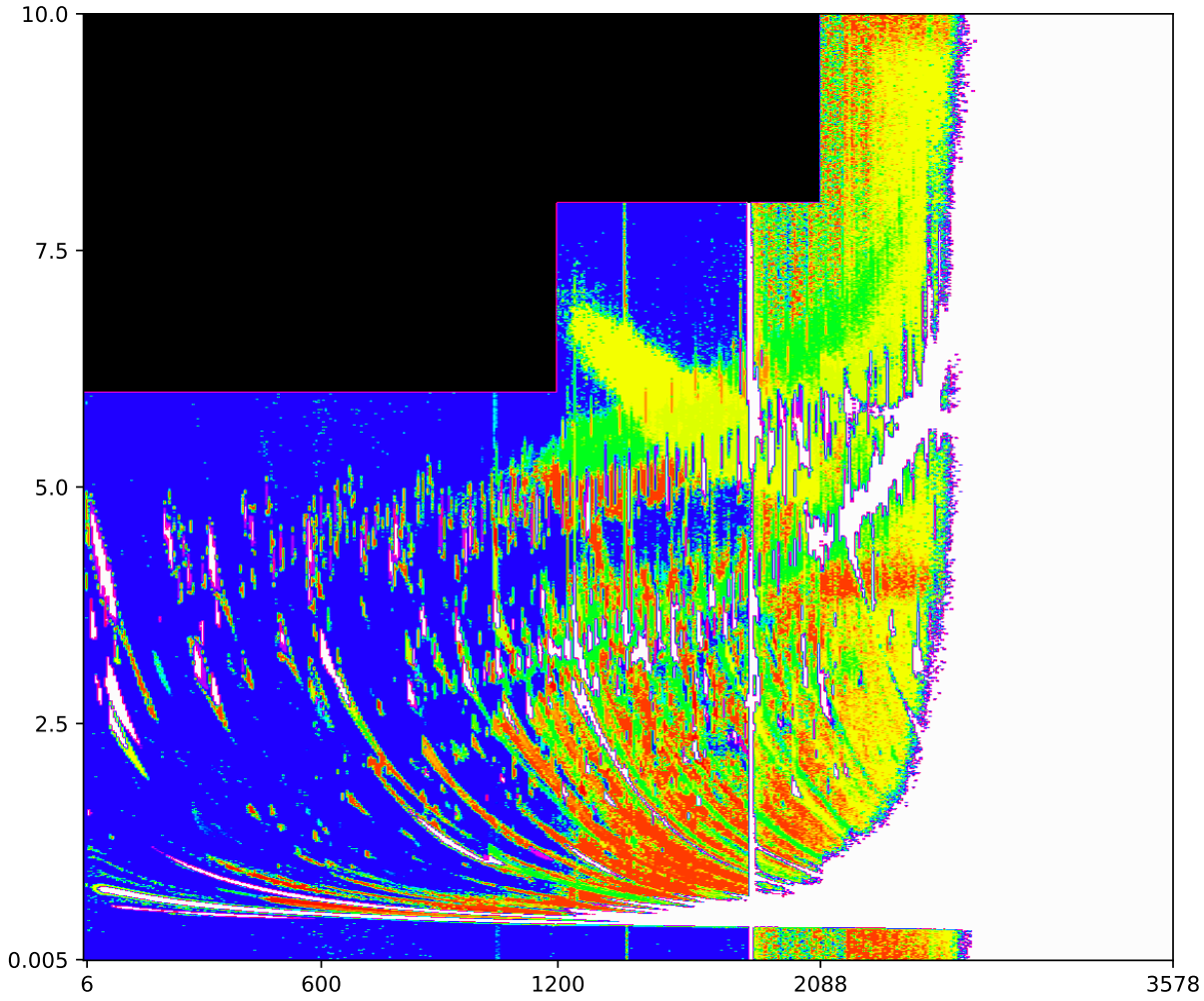


Figure A.9: Test sample, Mahalanobis OOD method, color coded predictions of gas chromatograph detector responses, predictions made without retention times, having 1200 ID samples classified as OOD samples by a threshold-based binary OOD classifier 5.10a

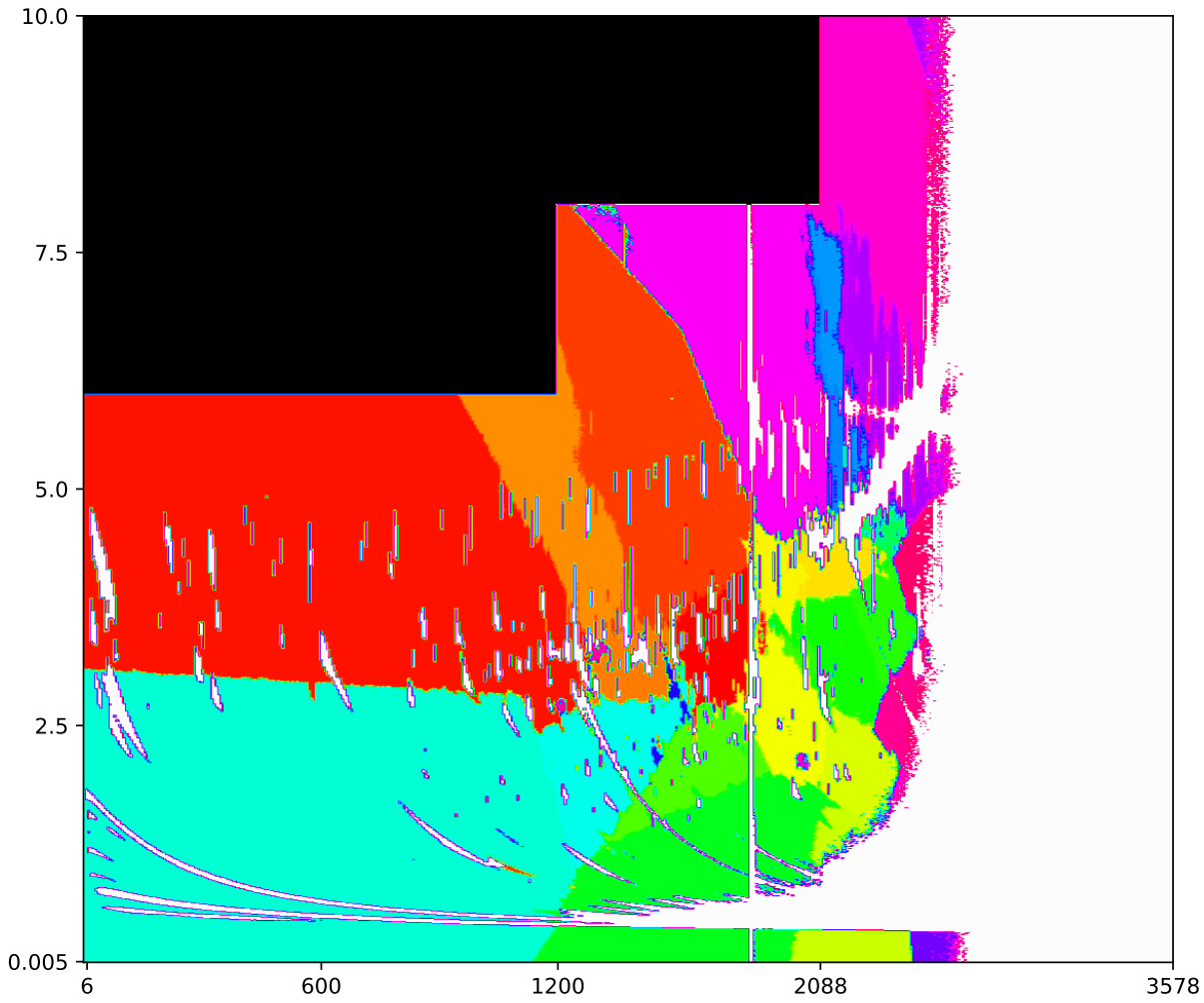


Figure A.10: Test sample, Mahalanobis OOD method, color coded predictions of gas chromatograph detector responses, predictions made with retention times, having 1200 ID samples classified as OOD samples by a threshold-based binary OOD classifier 5.10b

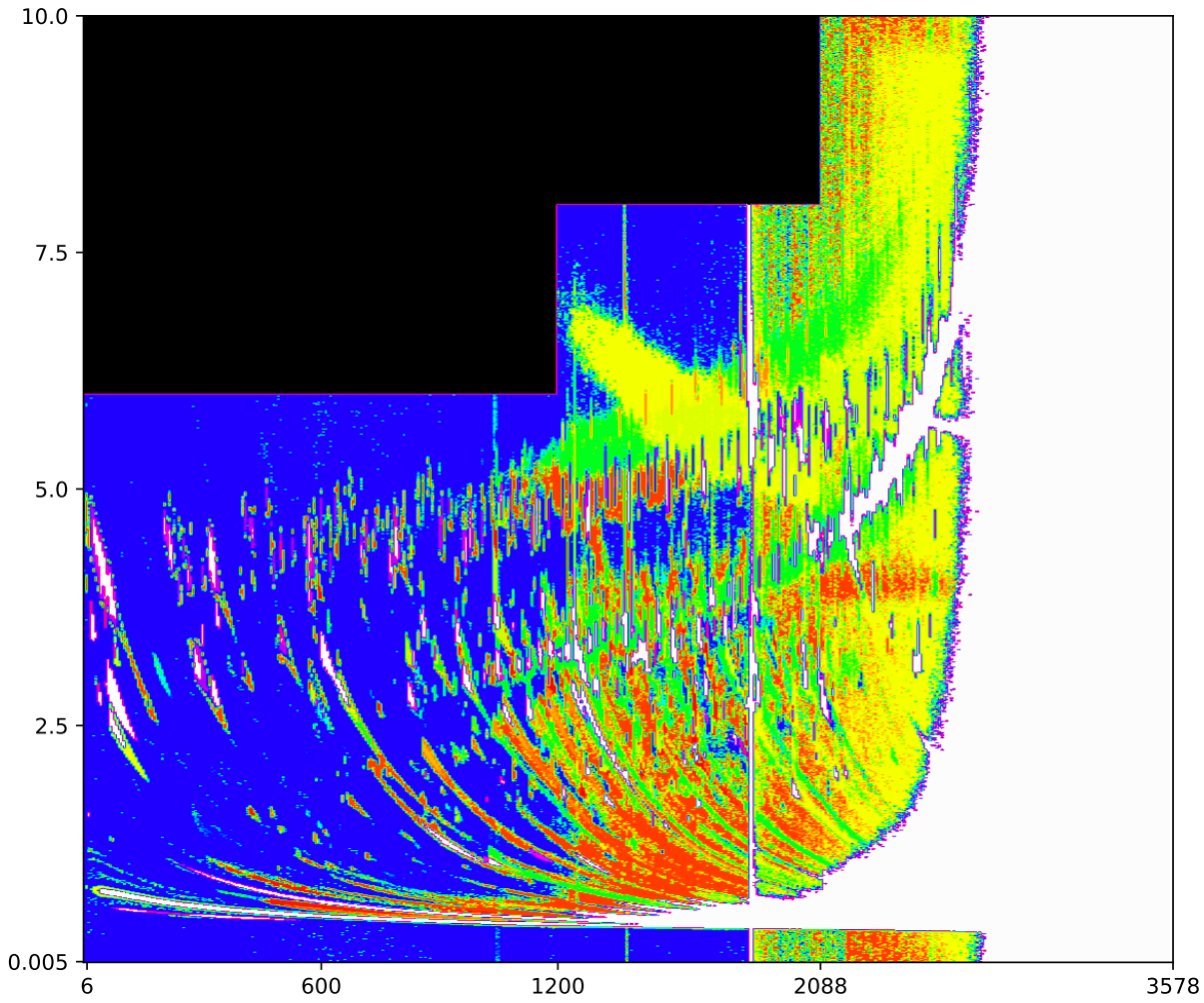


Figure A.11: Test sample, Mahalanobis OOD method, color coded predictions of gas chromatograph detector responses, predictions made without retention times, having 460 ID samples classified as OOD samples by a threshold-based binary OOD classifier 5.10c

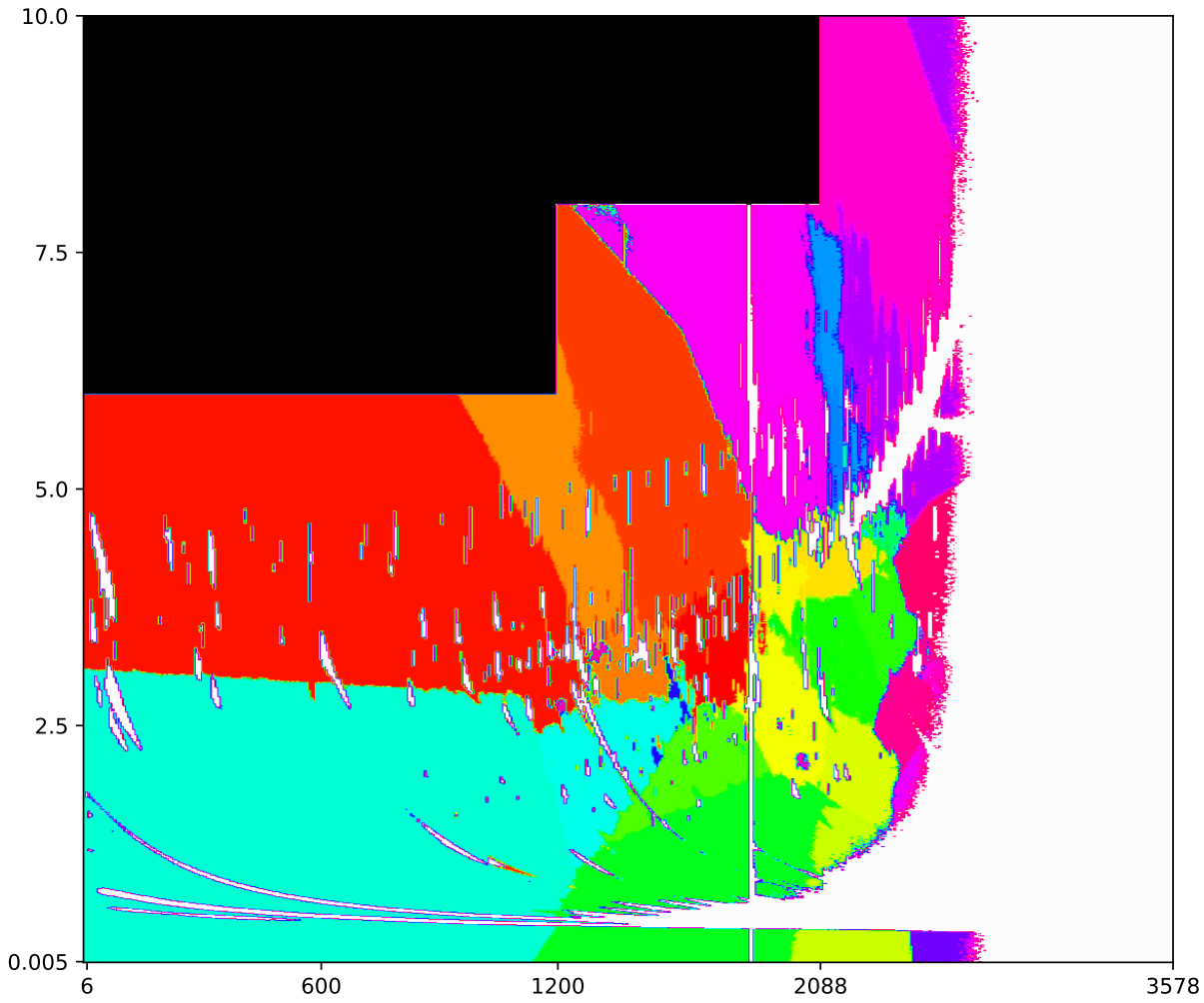


Figure A.12: Test sample, Mahalanobis OOD method, color coded predictions of gas chromatograph detector responses, predictions made with retention times, having 460 ID samples classified as OOD samples by a threshold-based binary OOD classifier 5.10d

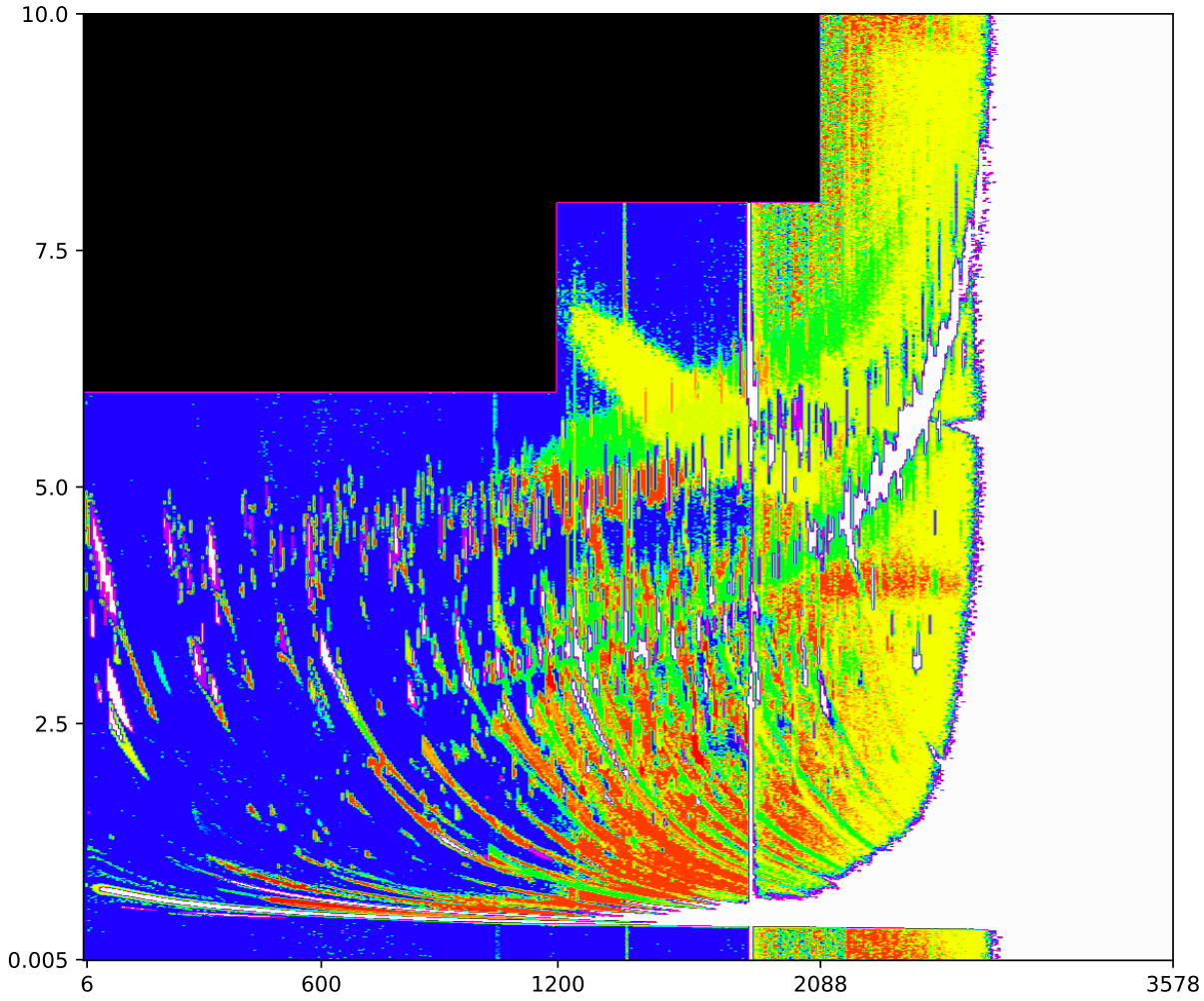


Figure A.13: Test sample, Mahalanobis OOD method, color coded predictions of gas chromatograph detector responses, predictions made without retention times, having 240 ID samples classified as OOD samples by a threshold-based binary OOD classifier 5.10e

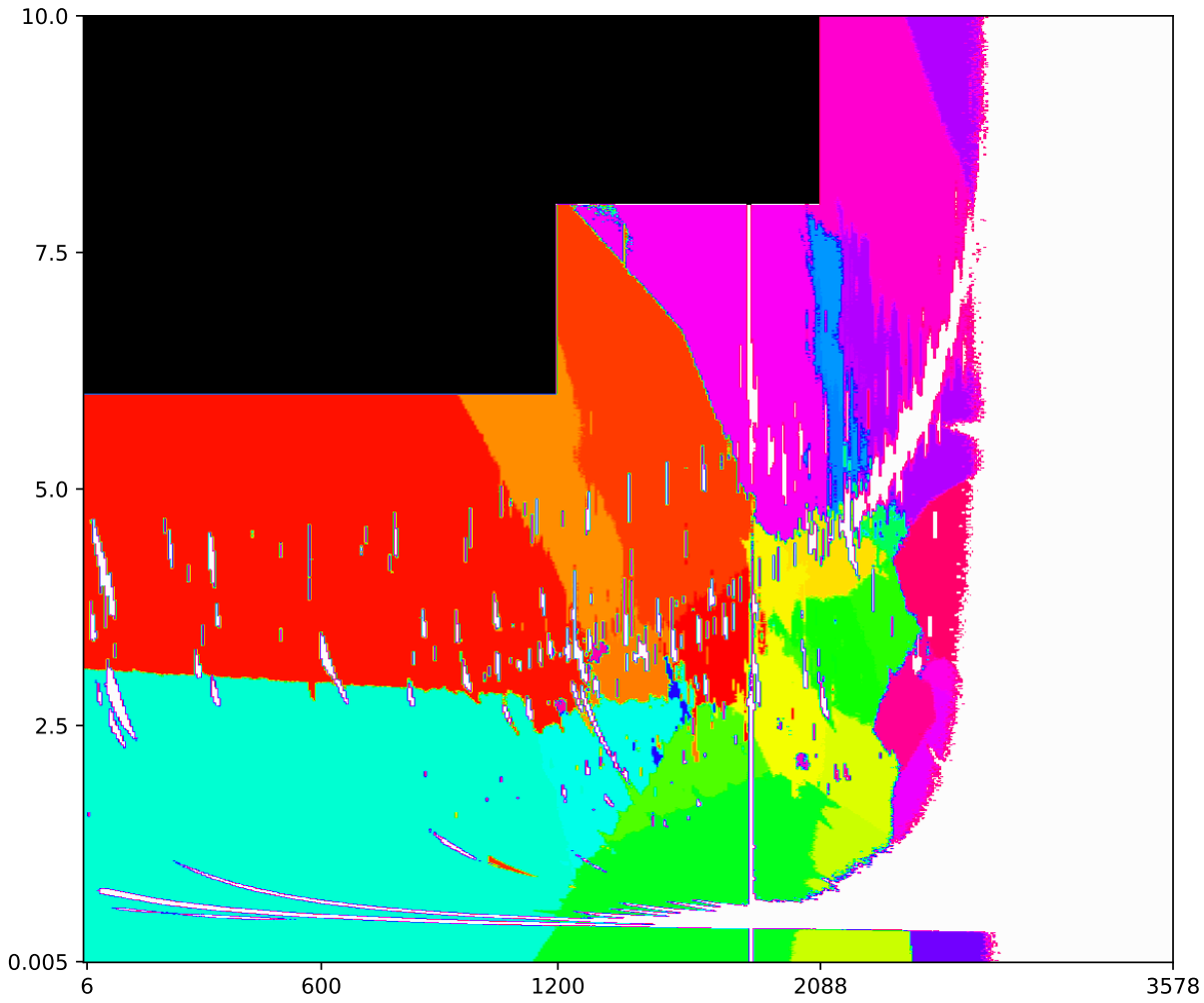


Figure A.14: Test sample, Mahalanobis OOD method, color coded predictions of gas chromatograph detector responses, predictions made with retention times, having 240 ID samples classified as OOD samples by a threshold-based binary OOD classifier 5.10f