Computers in Biology and Medicine 186 (2025) 109695

Contents lists available at ScienceDirect

Computers in Biology and Medicine

journal homepage: www.elsevier.com/locate/compbiomed

Pre-trained Maldi Transformers improve MALDI-TOF MS-based prediction

Gaetan De Waele [a,*], Gerben Menschaert [a], Peter Vandamme [b], Willem Waegeman [a]

[a] Department of Data Analysis and Mathematical Modelling, Ghent University, Coupure Links 653, Ghent, 9000, Belgium
[b] Laboratory of Microbiology, Ghent University, K. L. Ledeganckstraat 35, Ghent, 9000, Belgium

ARTICLE INFO

Keywords:
MALDI-TOF MS
Neural networks
Transformers

ABSTRACT

For the last decade, matrix-assisted laser desorption/ionization time-of-flight mass spectrometry (MALDI-TOF MS) has been the reference method for species identification in clinical microbiology. Hampered by a historical lack of open data, machine learning research towards models specifically adapted to MALDI-TOF MS remains in its infancy. Given the growing complexity of available datasets (such as large-scale antimicrobial resistance prediction), a need for models that (1) are specifically designed for MALDI-TOF MS data, and (2) have high representational capacity, presents itself.

Here, we introduce Maldi Transformer, an adaptation of the state-of-the-art transformer architecture to the MALDI-TOF mass spectral domain. We propose the first self-supervised pre-training technique specifically designed for mass spectra. The technique is based on shuffling peaks across spectra, and pre-training the transformer as a peak discriminator. Extensive benchmarks confirm the efficacy of this novel design. The final result is a model exhibiting state-of-the-art (or competitive) performance on downstream prediction tasks. In addition, we show that Maldi Transformer's identification of noisy spectra may be leveraged towards higher predictive performance.

All code supporting this study is distributed on PyPI and is packaged under: https://github.com/gdewael/maldi-nn.

1. Introduction

Matrix-assisted laser desorption/ionization time-of-flight mass spectrometry (MALDI-TOF MS) is a proteomic technique commonly used to identify microbial species [1]. First introduced to clinical microbiology at the end of 2010, its routine use is characterized by low-cost, speed, and reliability [2]. MALDI-TOF MS generates spectra containing peaks signifying mostly ribosomal proteins [3]. As such, the spectra can serve as fingerprints indicative of species identity [4].

For bacterial species identification, clinical diagnostic labs will typically use the solutions provided by MALDI-TOF MS manufacturers. These solutions are built on large, proprietary, in-house databases [5]. The models used in such solutions presumably rely on querying certain marker peaks to a large database [6]. This strategy works reasonably well for identification of most species, but some strains remain problematic to identify this way [7,8]. In addition, the peak-matching approach does not suit more-difficult prediction tasks such as strain typing [9], antimicrobial resistance prediction [10], and virulence factor detection [11]. In these cases, researchers turn to machine learning in order to possibly mine more-intricate patterns from the spectra.

Historically, machine learning for MALDI-TOF MS has been hampered by a lack of large open data. Because of this, the nascent field has not often progressed beyond off-the-shelf learning techniques [2]. Only a handful of examples exist of more advanced machine learning methods specifically adapted to a MALDI-TOF-based task [12–15]. As such, the design of machine learning algorithms specifically for MALDI-TOF mass spectra has not been studied yet in sufficient detail.

Typically, MALDI-TOF MS-based machine learning methods have either discretized the (m/z)-axis, or have pre-selected a number of peak locations as features based on training set characteristics. Discretizing the (m/z)-axis results in fixed-length spectral representations, where every feature consists of a bin. If bins are chosen too large, resolution is lost and multiple peaks may be lumped together. On the other hand, small bins result in a higher-dimensional representation where most bins contain no peaks, unnecessarily adding model complexity [13,14]. Other works proposed to select a fixed set of (m/z) locations based on training set characteristics [16]. Features for every spectrum are then derived by encoding the peak height (or binary peak presence) for every selected (m/z) location. The disadvantage here is that spectra may

* Corresponding author.
E-mail address: gaetan.dewaele@ugent.be (G. De Waele).

https://doi.org/10.1016/j.compbiomed.2025.109695
Received 18 January 2024; Received in revised form 10 January 2025; Accepted 13 January 2025
Available online 22 January 2025

contain important peaks not included in the fixed set of selected (m/z) locations. Hence, the model is unable to cope with patterns differing too much from general training set characteristics (e.g. rare peaks).

The information in MALDI-TOF mass spectra is defined by their peaks (many associated with ribosomal subunits) [17]. As such, we argue that, ideally, a machine learning model for MALDI-TOF mass spectra operates directly on sets of peaks as inputs. This, however, constitutes a non-trivial machine learning setup. A kernel-based technique that processes spectra in this way was previously proposed by Weis et al. [13]. However, their method is difficult to scale to larger sample sizes. Consequently, in this work, we have developed a deep learning-based solution operating on sets of peaks: a transformer for MALDI-TOF MS data.

Transformers are an ideal neural network candidate for learning on mass spectra for several reasons. Unlike MLPs and CNNs, they do not require a fixed-length feature vector, and hence do not require binning the (m/z)-axis. Further, unlike RNNs, they do not assume a sequentiality to the signal. Rather, transformers use a permutation-equivariant self-attention mechanism to perform message passing on a complete directed graph (i.e. peaks are nodes, and all nodes are connected to each other) [18]. As such, transformers can be viewed as operating on input sets, rather than sequences.

Because transformers can be viewed as operating on a complete digraph, they place no inductive bias on the learned patterns between input tokens. This property lends transformers supreme representational capabilities, but is also the reason why they are typically described as data-hungry. Consequently, transformers are usually mentioned in the same breath as self-supervised learning (SSL) [19]. SSL is the paradigm within deep learning wherein a supervised learning task is designed for data that has not explicitly been labeled [20]. This task is (usually) not a useful prediction problem in itself, but rather serves to pre-train a large model. In doing so, greater downstream performance on tasks of interest can be obtained. SSL is, therefore, most useful in scenarios where labeled data is limited, such as the MALDI-TOF MS domain [2].

As related works, following their rise to dominance in natural language processing, transformers are increasingly adopted towards biological data modalities [21–24]. Recently, they have also been applied to the mass spectral data domain for (1) *de novo* peptide sequencing [25,26], and (2) annotating metabolite mass spectra [27,28]. In the latter data domain, researchers have successfully used the masked language modeling paradigm [29] to pre-train transformers towards better performance [30].

In the present study, we have developed Maldi Transformer, a deep learning architecture for processing MALDI-TOF mass spectra. The inputs to Maldi Transformer consist of spectra as sets of peaks. To fully take advantage of the transformer architecture's representational capacity, we have proposed a novel self-supervised pre-training strategy. Our strategy relies on discriminating real peaks from noisy ones introduced to the spectrum via shuffling peaks across samples. We pre-trained Maldi Transformer on the large open DRIAMS database [2]. Maldi Transformer obtains strong performance on downstream benchmarks, demonstrating the power of the approach.

2. Methods

To train and test models, we use three large (>10 000 spectra) MALDI-TOF datasets, two of which are available to the public domain (Section 2.1). We define a transformer for MALDI-TOF MS data, and design a novel pre-training task (Section 2.2). We experimentally validate the final architecture on a number of downstream tasks (Section 2.3). All models and experiments are implemented using PyTorch and are publicly available under https://github.com/gdewael/maldi-nn. The following sections outline each component of our study in detail.

Table 1
Data sources used, along with their use and sizes (in number of spectra).

Dataset	Used for	Train-Val-Test size
DRIAMS	Pre-training	207 172 - 21 440 - 21 443[a,b]
DRIAMS-A	Fine-tuning on AMR prediction	28 331 - 4994 - 4999[c]
RKI	Fine-tuning on species identification	8442 - 1350 - 1263
LM-UGent	Fine-tuning on species identification	88 267 - 8710 - 8700

[a] Contains all spectra in DRIAMS with at least 200 detected peaks.

[b] Of which 97 783, 14 055, and 14 183 have species labels in train, val and test splits, respectively.

[c] Numbers reflect spectra. In total, 409 395, 76 431, and 76 133 AMR labels across 65 drugs are associated with those splits.

2.1. Data

A first dataset is the recently published DRIAMS database [10]. It is used to both pre-train Maldi Transformer and to fine-tune it on antimicrobial resistance (AMR) prediction (binary classification using a dual-branch recommender system). DRIAMS contains a total of 250 070 spectra, originating from four hospitals in Switzerland. For AMR prediction, the same data splits are used as in earlier work [15]. Briefly, DRIAMS-A spectra from before 2018 are split to the training fraction, whereas DRIAMS-A spectra measured during 2018 are evenly split between validation and test set. For pre-training, the same splits are used, but all spectra from DRIAMS-B, -C, and -D are additionally added to the training set. Apart from AMR measurements, DRIAMS contains species labels for many spectra. These labels are derived from the species identification pipelines included with the MALDI-TOF MS machines. After processing (see Appendix A), the pre-training DRIAMS dataset spans 469 species.

The second dataset consists of the public Robert Koch-Institute (RKI) database [31]. The final used dataset is a private historical database containing more than 2400 taxonomic reference strains, cultured and analyzed by the Laboratory of Microbiology at Ghent University. In this manuscript, both datasets will be abbreviated as RKI and LM-UGent, respectively. The RKI dataset contains MALDI-TOF mass spectra from highly pathogenic bacteria, covering similar species as in DRIAMS. The LM-UGent dataset, on the other hand, includes a broader taxonomic range. Both datasets are used for fine-tuning on species identification (multi-class classification). As in [14], in order to create a challenging training-validation-test split, spectra are split in such a way that there is no overlap in terms of strains. As a consequence, the models are tested whether they can identify unseen strains of (seen) species. The following rules are used in data splitting: all spectra for a species are assigned to the training set if that species only has one strain in the dataset. If the species has more than one strain, strains are split such that 80% of strains of that species are assigned to the training set, and the other 20% evenly split between validation and test set (with a floor value of at least one strain being assigned to either validation or test set). The total number of species in the RKI training set spans 270, of which 106 and 108 are present in the validation and test set, respectively. For the LM-UGent dataset, these numbers are 1088, 200, and 202, for the training, validation and test set, respectively. For more details on the LM-UGent dataset, the reader is referred to [14]. Table 1 lists a summary of the sizes of all used data.

All MALDI-TOF mass spectra are preprocessed using standard practices [32]. Following [10], all spectra undergo the following steps: (1) square-root transformation of the intensities, (2) smoothing using a Savitzky-Golay filter with half-window size of 10, (3) baseline correction using 20 iterations of the SNIP algorithm, (4) trimming to the 2000–20,000 Da range, (5) intensity calibration so that the total intensity sums to 1. For Maldi Transformer, peaks are then detected on the preprocessed spectrum using the persistence transformation, introduced by [13]. While the original publication proposes this algorithm to nullify other preprocessing steps, we find that prior preprocessing

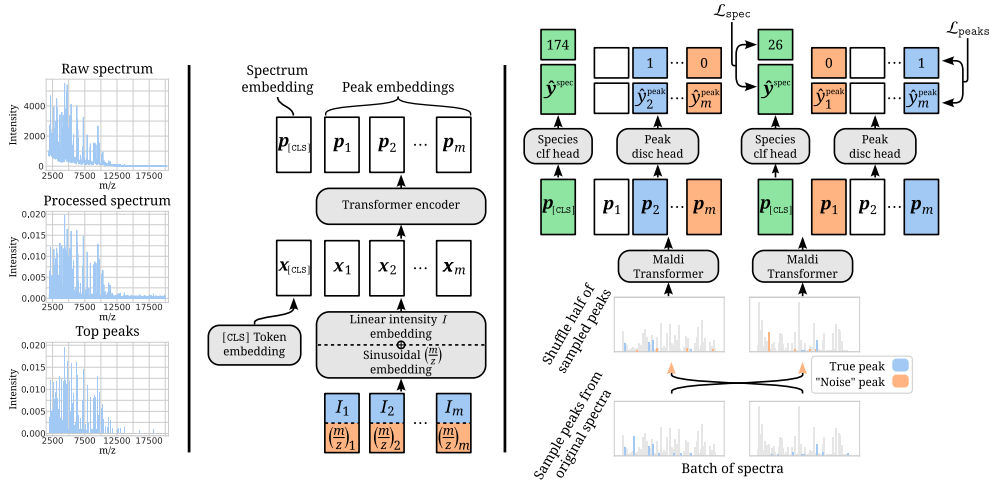

Fig. 1. Left: Representation of MALDI-TOF mass spectra. A raw spectrum is preprocessed, after which a topological persistence transformation is performed to detect peaks, resulting in a sparse representation of the spectrum [13]. **Middle:** Maldi Transformer. A peak is characterized by an intensity I and (m/z) value. Both are embedded to higher dimensional space by a linear layer and sinusoidal embedding, respectively, and are then summed. A [CLS] token is prepended and the resulting vectors $\boldsymbol{x}$ are sent through multiple Transformer encoder layers. The resulting output vectors $\boldsymbol{p}_{j\in\{1,\dots,m\}}$ can be used as peak embeddings. Additionally, $\boldsymbol{p}_{[\text{CLS}]}$ can be used as a summary embedding of the whole spectrum. **Right:** Proposed peak discrimination pre-training strategy. In a mini-batch of spectra, 15% of peaks are randomly sampled, half of which are shuffled among all spectra in the batch. The resulting spectra are encoded with Maldi Transformer. The resulting peak embeddings $\boldsymbol{p}_{j\in\{1,\dots,m\}}$ are sent through a linear layer ("Peak disc head") trying to distinguish original peaks (blue) from shuffled "noise" peaks (orange). Spectrum embeddings $\boldsymbol{p}_{[\text{CLS}]}$ are sent through a separate linear output layer ("Species clf head") to predict species identity (green). The numbers 174 and 26 refer to example ground-truth class indices $c = y^{\text{spec}}$ (indicating the microbial species of the spectrum).

steps help peak detection (see Appendix B). As in [13], we select the highest 200 peaks for every spectrum as inputs.[1] Maldi Transformer is compared against baseline methods, which require a fixed-length input. For these models, instead of running peak detection as a last preprocessing step, spectra are instead binned to a 6000-dimensional vector by summing together intensities in intervals of 3 Da.

2.2. Maldi Transformer

Model. To introduce Maldi Transformer, let us denote a spectrum as a set of m peaks $S = \left\{\left((m/z)_j, I_j\right)\right\}_{j=1}^{m}$, with each peak characterized by its (m/z) value and (preprocessed) intensity I. An annotated dataset $\mathcal{D}$ then consists of a set of n samples $\mathcal{D} = \left\{\left(S_i, y_i^{\text{spec}}\right)\right\}_{i=1}^{n}$, with y^{spec} the species label.

Maldi Transformer requires a single input representation $\boldsymbol{x} \in \mathbb{R}^d$ for each peak $\left((m/z)_j, I_j\right)$. To achieve this, intensities I are linearly transformed to a d-dimensional space. Similarly, (m/z) values are embedded to sinusoidal positional encodings [18]. For any (m/z) value, the positional encoding (PE) at feature index f is

$$PE_{((m/z),f)} = \begin{cases} sin\left(\frac{(m/z)}{10 \cdot 10000^{f/d}}\right), & \text{if } f \text{ is even} \\ cos\left(\frac{(m/z)}{10 \cdot 10000^{(f-1)/d}}\right), & \text{if } f \text{ is odd} \end{cases}$$

Note that this formulation is identical to the one described in [18], apart from the factor 10 division of the (m/z) positions. This division is performed to bring the numerical range of (m/z) values (2000 Da - 20,000 Da) closer to the numerical range of positional indices for which this equation was originally designed.[2]

The sinusoidal (m/z) embedding and linear intensity I embedding are summed to a single input representation per peak $\boldsymbol{x}_{j\in\{1,\dots,m\}}$. A trainable [CLS] vector is prepended to the input for spectrum-level prediction, following common practice [29,34]. The final input to the encoder-only transformer is, hence, $\boldsymbol{X} \in \mathbb{R}^{201\times d}$, with 201 the number of peaks plus the [CLS] token, and d the hidden dimensionality of the model. The encoder-only transformer processes $\boldsymbol{X}$ to a spectrum-level output embedding $\boldsymbol{p}_{[\text{CLS}]}$ and output peak-level embeddings $\boldsymbol{p}_{j\in\{1,\dots,m\}}$. The design of the transformer encoder blocks follows current state-of-the-art practices (Fig. E.6) [35]. An overview of the model is visualized in Fig. 1.

Pre-training task design. To boost the performance of Maldi Transformer on supervised tasks with limited labeled data, a novel self-supervised pre-training strategy is designed. We propose to pre-train Maldi Transformers as peak discriminators. That is, in a batch of spectra, some peaks are randomly sampled to use for training. Following Devlin et al. [29], we sample 15% of the peaks. Half of those sampled peaks are shuffled among all spectra, while the others are kept as part of their original spectrum. Using the shuffled peaks as negative "noise" peaks in a spectrum, a discriminative model is trained to distinguish the noise peaks from the sampled original ones using the cross-entropy loss.

More formally, for a spectrum S, let us denote its version with shuffled peaks as S^{shuff}. Further, let $\boldsymbol{j}_{\text{train}} = \{j_1, \dots, j_s\}$ denote the indices of its sampled peaks. Maldi Transformer processes S^{shuff} to output peak-level embeddings $\boldsymbol{p}_{j\in\{1,\dots,m\}}$, which are then processed to predictions $\hat{y}^{\text{peak}}_{j\in\{1,\dots,m\}} = \sigma\left(\boldsymbol{w}_p^{\intercal}\boldsymbol{p}_{j\in\{1,\dots,m\}}\right)$, with $\boldsymbol{w}_p$ a learnable weight vector. The peak discrimination loss is, then, given by

$$\mathcal{L}_{\text{peaks}} = \frac{1}{|\boldsymbol{j}_{\text{train}}|} \sum_{j_{\text{train}}} \Big(-\mathbb{I}\left(S_j = S_j^{\text{shuff}}\right) \log\left(\hat{y}_j^{\text{peak}}\right) - \mathbb{I}\left(S_j \neq S_j^{\text{shuff}}\right) \log\left(1 - \hat{y}_j^{\text{peak}}\right) \Big)$$

with $\mathbb{I}$ the indicator function, and j indexing single peaks from the spectrum S.

Complementary to the peak discrimination strategy, the spectrum embedding $\boldsymbol{p}_{[\text{CLS}]}$ is sent to a multi-class linear output head to predict the microbial species identity y^{spec} of the original spectrum[3]:

[1] Note that Maldi Transformer can, in principle, deal with variable number of peaks per spectrum.

[2] Language transformers are often trained with maximum sequence lengths between 512 and 2048 [33].

[3] While this is not a purely self-supervised training objective, we justify its use with two reasons. First, in language, pre-training with a sentence-level [CLS] task boosts downstream performance [29]. Second, due to the integrated nature of MS manufacturers' species identification pipelines, bacterial species labels are relatively easy to come by. As such, no manual labeling is necessary to obtain these labels. Additionally, it has to be noted that, as not all spectra in DRIAMS carry a species label, the species classification loss is only calculated for labeled spectra.

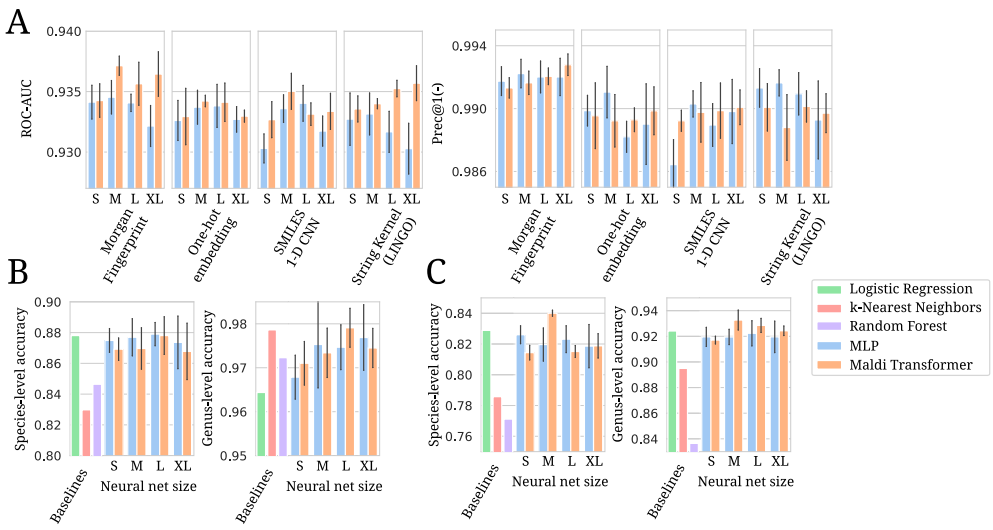

Fig. 2. Barplots of all main experimental results. Error-bars indicate standard deviation over five independent runs. Neural network model sizes range from S to XL. Details per model size for Maldi Transformer and MLP models are listed in Tables 2 and D.3, respectively. **A:** AMR prediction results on DRIAMS. Performances are shown for models using four different drug embedders in a dual-branch recommender system. **B:** Species identification results on the RKI dataset. **C:** Species identification results on the LM-UGent dataset.

Table 2
Pre-training configurations for different Maldi Transformer sizes. Species clf λ refers to the probability that the species identification loss is applied per training step.

Hyperparameter	Model			
	S	M	L	XL
# params	1.65M	3.27M	6.92M	14.84M
# layers	4	6	8	10
Hidden dim d	160	184	232	304
# heads	8	8	8	8
Learning rate	5e−4	5e−4	5e−4	3e−4
Pre-training steps	500 000	500 000	500 000	400 000
Species clf λ	Bern(0.01)	Bern(0.01)	Bern(0.01)	Bern(0.005)

$\hat{\boldsymbol{y}}^{\text{spec}} = \texttt{Softmax}(\boldsymbol{W}_s^{\intercal} \cdot \boldsymbol{p}_{[\texttt{CLS}]})$, with $\boldsymbol{W}_s$ a learnable weight matrix. Species predictions $\hat{\boldsymbol{y}}^{\text{spec}}$ are optimized with multi-class cross-entropy: $\mathcal{L}_{\text{spec}} = -\log \hat{y}_c^{\text{spec}}$, with $c = y^{\text{spec}}$ the ground-truth class index. The final pre-training loss function is the sum of peak discrimination binary cross-entropy and species identification multi-class cross-entropy: $\mathcal{L} = \mathcal{L}_{\text{peaks}} + \lambda\, \mathcal{L}_{\text{spec}}$, with $\lambda \sim \text{Bern}(0.01)$. The species identification loss $\mathcal{L}_{\text{spec}}$ is only randomly applied in 1% of the training steps. This is performed because, empirically, overfitting of the species classification task is observed when applied at every training step (see Fig. E.7). A visual representation of the entire pre-training approach is shown in Fig. 1. A more-thorough description of the whole pre-training strategy can be found in Appendix C Algorithm 1.

Our novel peak discrimination pre-training strategy is proposed due to conceptual issues with porting established pre-training approaches to the MALDI-TOF MS domain, a point further elaborated on in Appendix C. We benchmark this novel pre-training strategy against alternatives in Section 3.2.

Model configurations. We train Maldi Transformer in four different sizes: S, M, L, and XL. Model sizes are chosen so that the total weight numbers roughly correspond to the ones in De Waele et al. [15]. Table 2 lists the size of all models, along with some hyperparameter settings. The Adam optimizer is used to pre-train all models in BFloat16 mixed precision [36]. Gradients are clipped to a norm of 1. A batch size of 1024 is applied for all models. A linear learning rate warm-up is applied over the first 2500 steps, after which the learning rate remains constant (Table 2). During training, a dropout of 0.2 is applied in the GLU feedforward and over the attention matrix.

2.3. Experimental set-up of downstream tasks

As mentioned in Section 2.1, Maldi Transformer's performance is validated on three downstream supervised tasks. For all three tasks, the pre-trained model is plugged in at initialization and all weights are fine-tuned (i.e. no weight freezing). A task-specific linear output head $\boldsymbol{W}_{out}$ projecting the spectrum embedding $\boldsymbol{p}_{[\texttt{CLS}]}$ to the desired output space is trained from scratch. For AMR prediction, Maldi Transformer is used as a spectrum embedder in a dual-branch neural network recommender system. To compare its performance against previous results, recommenders are trained with the four best-scoring drug embedders in [15]. To benchmark Maldi Transformer in terms of species identification, the RKI and LM-UGent datasets are used. Species identification is compared to MLP baselines, Logistic Regression, Random Forest, and k-nearest neighbors (k-NN) models. Details on the exact fine-tuning setups for Maldi Transformer, as well as training details for all baselines are found in Appendix D.

The LM-UGent dataset covers a broader species diversity than the clinical species found in the pre-training set. As such, this dataset can be considered out-of-distribution for the pre-trained Maldi Transformer. For this reason, a domain adaptation step on the pre-trained Maldi Transformer is performed before supervised fine-tuning. The domain adaptation step consists of pre-training Maldi Transformer for 20 000 additional steps in the same fashion, but now using the LM-UGent dataset, instead of DRIAMS.[4]

3. Results

In the following sections, we first examine Maldi Transformer's predictive performance compared to baselines (Section 3.1). We experimentally validate the soundness of our model design through ablations (Section 3.2). Finally, we exploit Maldi Transformer's denoising properties in Section 3.3.

[4] By performing domain adaptation, in contrast to the other supervised tasks, the task-specific output head $\boldsymbol{W}_{out}$ does not need to be initialized from scratch anymore, but can be copied from the pre-trained model, as it already has the right dimensions.

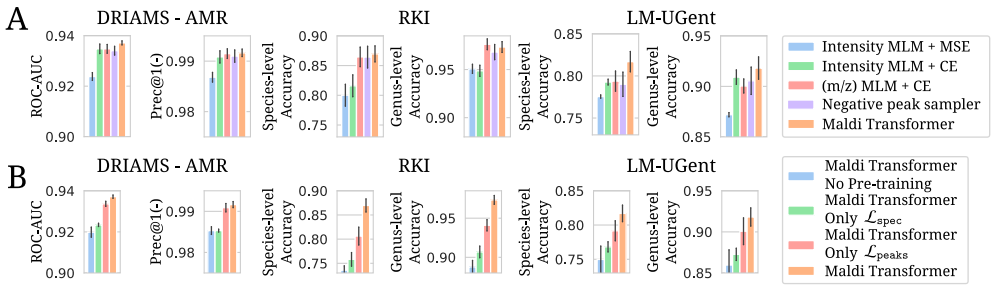

Fig. 3. Barplots of all ablation results. Only results for Medium-sized transformers are shown. The final model as discussed in Section 2.2 is labeled as Maldi Transformer. Error-bars indicate standard deviation over five independent runs. For DRIAMS AMR prediction, only results using a Morgan Fingerprint drug embedder are shown. For the LM-UGent dataset, results are shown without the domain adaptation step, as this step was also not performed for ablation models. See Fig. E.13 for the effects of this step. **A:** Testing different ways of pre-training a transformer on MALDI-TOF MS data. **B:** Leaving out different parts of the final pre-training strategy.

3.1. Maldi Transformer improves performance on downstream tasks

Pre-training curves for Maldi Transformer are shown in Fig. E.8. Maldi Transformer spectrum embeddings after pre-training are visualized in Fig. E.9. After pre-training, Maldi Transformer is fine-tuned w.r.t. a downstream task. Maldi Transformer's downstream performance is compared to preprocessed and binned baselines. For AMR prediction, it is compared to MLPs of similar sizes. For species identification, it is additionally compared to non-neural network baselines.

Fig. 2 shows the experimental results for all downstream tasks. For AMR prediction, models are evaluated in terms of their (per-spectrum-average) ROC-AUC and Precision@1 of the negative class (Prec@1(-)).[5] Fig. E.10 additionally shows their performance in terms of (drug) macro ROC-AUC. For species identification, models are evaluated in terms of species- and genus-level accuracy. In general, Maldi Transformer obtains superior performance in comparison to other tested methods. For AMR prediction, Maldi Transformer consistently outperforms all MLP models in terms of ROC-AUC. In terms of the Prec@1(-), Maldi Transformer is sometimes outperformed by MLP models, but the best-scoring model overall is still one using Maldi Transformer (i.e. the XL model paired with a Morgan Fingerprint drug embedder). On drug Macro ROC-AUC (Fig. E.10), Maldi Transformer consistently outperforms MLPs. As this metric computes how well the model separates spectra in terms of their resistance status per drug, this metric is especially apt for showing the advantage of pre-t raining to improve spectral representations.

For species identification on the RKI dataset, Maldi Transformer does not always provide a clear advantage in terms of species-level accuracy. For genus-level accuracy, however, the large Maldi Transformer variant achieves the best performance, with a k-NN model providing competitive results.

On the larger and more-difficult LM-UGent dataset, results are again more convincing. The best-performing species-level model is the medium-sized Maldi Transformer, convincingly beating other models with 84% accuracy. The same results are obtained on genus-level accuracy, where Maldi Transformer consistently beats other models.

In order to gain a better understanding into when Maldi Transformer might provide advantages over other models, an experiment is run on the LM-UGent dataset. The experiment entails progressively leaving out species (classes), keeping only the top-n most occurring species, and training and evaluating models on that subset. The results, visualized in Fig. E.12, show that by gradually decreasing the number of classes, and, hence, the difficulty of the prediction task, Maldi Transformer's advantage on LM-UGent gradually disappears.

[5] The spectrum-macro ROC-AUC evaluates the average ranking quality of recommended drugs. The Precision at 1 of the negative class evaluates the proportion of top-recommended drugs that are correct. These metrics are described in more detail in De Waele et al. [15].

3.2. Maldi Transformer pre-training ablation

To further validate the efficacy of the proposed Maldi Transformer, its performance is compared against four alternative realizations of transformers on MALDI-TOF MS data (i.e. using four alternative pre-training strategies). The first takes inspiration from the masked language model (MLM) BERT [29]. Here, intensity values of peaks are randomly masked out, and a transformer is pre-trained to predict the original values back using the mean-squared error loss. In the second, the same procedure is followed, but instead of regressing masked intensities, on the output-level, intensities are binned into discrete height bins. A model is then pre-trained to classify the bin of the masked intensity using the cross-entropy loss. The third pre-training strategy consists of the same as the previous, but masking (m/z) values instead, much as like proposed in Bushuiev et al. [30]. In the last, a discriminative model much like our final proposed strategy is trained. The difference in this model is that negative peaks are sampled from some estimated distribution of peaks, instead of generating negative peaks by shuffling. All alternative pre-training strategies are described in greater detail in Appendix C. In Fig. 3A, it is observed that our proposed peak shuffling pre-training strategy consistently outperforms competing approaches. Our results show merit to the design of a pre-training task specifically adapted to the MS domain. In comparison, MLM-based designs ported from the language domain result in comparatively worse performance.

Fig. 3B shows ablation results when leaving different parts out of the final pre-training strategy. Maldi Transformer is compared to models where one of the two loss components are left out: either the peak discrimination loss ($\mathcal{L}_{\text{peaks}}$), or the species identification loss ($\mathcal{L}_{\text{spec}}$). It is also compared to a transformer model without pre-training whatsoever. It can be seen that the latter strategy (i.e. training a separate transformer from scratch for every supervised task), delivers subpar performance. In addition, only pre-training on species identification does not help much. The biggest gain is made from the peak discrimination task. Yet, overall, the effects are additive, showing that each part contributes to the final performance.

A final ablation is to test the choice of peak detection. Here, the persistence transformation algorithm [13] used in Maldi Transformer is tested against the established MALDIquant approach to detect peaks [32]. The results of this ablation are shown in Fig. E.11.

3.3. Maldi Transformer model analysis

Due to its pre-training design, Maldi Transformer outputs the probability of a peak belonging to its spectrum for every peak. More formally, let us denote the output probability of a peak j belonging to its spectrum S as $\Pr\left(y_j^{\text{peak}} = 1|S\right)$. This quantity can be obtained through the pre-trained Maldi Transformer: $\Pr\left(y_j^{\text{peak}} = 1|S\right) = \hat{y}_j^{\text{peak}} = \sigma(\boldsymbol{w}_p^{\text{I}} \cdot \boldsymbol{p}_j)$. During pre-training, as peaks are shuffled between spectra, one expects to encounter both positives and negative true labels. In this section, we examine how peak output probabilities $\Pr\left(y_j^{\text{peak}} = 1|S\right)$ may be used

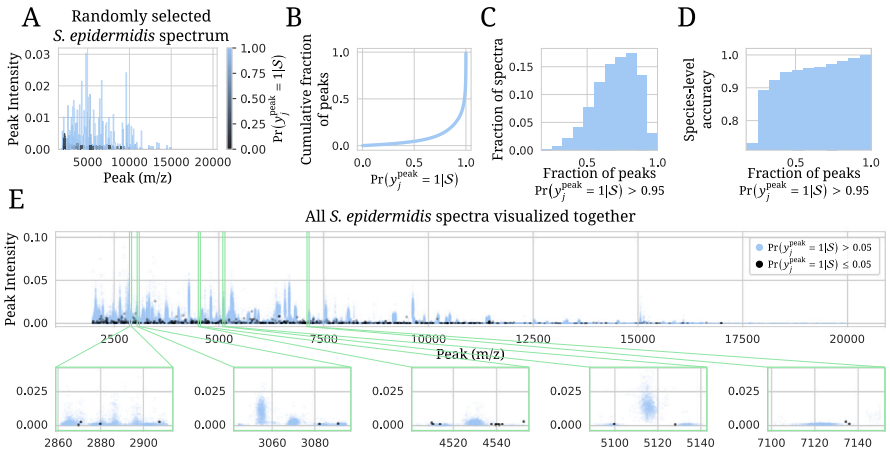

Fig. 4. (Medium-sized) Maldi Transformer analysis on DRIAMS pre-training test data. Note that no shuffling of peaks across spectra is performed to generate these plots. Each spectrum contains their original detected peaks. **A:** A randomly selected spectrum, each peak colored by the model's output probability of that peak belonging to that spectrum or not $\Pr\left(y_j^{\text{peak}} = 1|S\right)$. **B:** Empirical cumulative distribution of those output probabilities over all spectra in the DRIAMS test set. **C:** Histogram of fraction of peaks confidently (>0.95) predicted as "true" peaks per spectrum. **D:** Species-level accuracy in function of fraction of peaks confidently (>0.95) predicted as "true" peaks per spectrum. **E:** All *S. epidermidis* spectra visualized together, every peak as a separate dot. Peaks confidently predicted as "noise" (≤0.05) are shown in black. Zoomed in subplots show that "noise" peaks originate outside clusters of usual peak locations.

during inference (i.e. without shuffling) using the medium-sized Maldi Transformer. The interpretation of probabilities requires a model to be calibrated. This property is further examined in Fig. E.14. Results in the following paragraphs are derived from the DRIAMS pre-training test set.

Maldi Transformer denoises spectra. In Fig. 4A, a randomly selected *Staphylococcus epidermidis*[6] spectrum is visualized, each peak colored according to its output probability $\Pr\left(y_j^{\text{peak}} = 1|S\right)$. Most peaks are (correctly) assigned a high probability of belonging to their spectrum. The peaks with a low probability could be mistaken by the model, or could represent "true" noise in the spectrum. Such noise may still be present in the final spectrum due to, for example (1) noisy readouts from the spectrometer, or (2) shortcomings in preprocessing. In order to validate this hypothesis, it makes sense to look at patterns across multiple spectra. In Fig. 4E, all *S. epidermidis* spectra are visualized together, each peak as a single dot. Black dots represent peaks that are predicted to be noise with high probability ($\Pr\left(y_j^{\text{peak}} = 1|S\right) \leq 0.05$). In the magnified parts of the plot, it can be seen that blue dots cluster together, meaning that *S. epidermidis* spectra often have peaks in the same places. Black dots mainly fall outside or on the edges of those clusters, signifying that those peaks are rightly picked up by the model as noise. Consequently, Maldi Transformers can serve a broader purpose as spectrum denoisers, in addition to their supervised learning capabilities.

Noisy peaks are indicative of lower downstream performance. In order to better grasp the effect of noisy peaks on spectra, peak predictions are examined across all spectra in the DRIAMS test set. Fig. 4B shows the empirical cumulative distribution of peak output probabilities. Only approximately 5% of all peaks are assigned a probability of belonging to their respective spectrum $\Pr\left(y_j^{\text{peak}} = 1|S\right)$ of lower than 50%. Conversely, half of the peaks are predicted with a probability of 98.5% or greater. This shows that while noisy peaks do exist, they typically make up a small percentage of the overall input spectrum.

Noisy peaks may not uniformly occur across the dataset. Some spectra may have more noisy peaks than others. A good spectrum quality statistic can, hence, be the fraction of peaks that are confidently predicted as "true", e.g. with a probability greater than 95%. Fig. 4C shows the distribution of spectra in function of this statistic. A slight left tail in the distribution signifies that some spectra have a large amount of noisy peaks. Approximately 10% of the spectra have more than half of their peaks predicted with a probability smaller than 95%. Fig. 4D shows that this statistic is also indicative of predictive performance. Spectra with more confidently "belonging" peaks have a higher species-level accuracy (on DRIAMS test set labels). Species-level accuracy ramps up from 90% (or lower) for spectra with a lot of noisy peaks, up to nearly-perfect accuracy for spectra with (almost) all "predicted true" peaks.[7] The fraction of "predicted true" peaks also correlates with prediction certainty (see Fig. E.15). These results showcase Maldi Transformer's ability to not only improve performance, but also provide further insights into the data.

4. Discussion

Maldi Transformer adapts the transformer model for MALDI-TOF MS data, and proposes to pre-train said model using a novel peak shuffling approach. Using this novel self-supervised task design, we show state-of-the-art (or competitive) performance on the most important real-world use-cases of MALDI-TOF MS. Importantly, we show that our novel self-supervised pre-training task design delivers better performance compared to competing pre-training paradigms.

Experiments on increasingly simpler species identification tasks (Fig. E.12) shows that the advantage of Maldi Transformer is most pronounced on more difficult tasks. This perhaps explains why experimental results on the smaller RKI dataset do not clearly show an advantage for Maldi Transformer. On the comparatively more complex tasks, such as 1088-way classification on LM-UGent and dual-branch recommendation on DRIAMS, Maldi Transformer consistently delivers state-of-the-art performance.

As with many subfields of machine learning, size, quality, and diversity of data constitute the biggest determinant of performance. In this study, an apparent ceiling in representational capacity has been obtained given the available public data. This is evidenced by the fact that embeddings of increasingly larger variants of Maldi Transformer do not uncover more fine-grained clusters (Fig. E.9). In addition, in

[6] *Staphylococcus epidermidis* is the most occurring species in the DRIAMS pre-training test set.

[7] Note that DRIAMS species labels are produced from the MS manufacturers' software and models. It can be expected that it is relatively easy for a model to reproduce the labels (predictions) from another model. Hence, the nearly-perfect species-level accuracy on DRIAMS is not unexpected.

fine-tuning, XL variants of Maldi Transformer typically do not give advantages over the M or L models. The size of the XL Maldi Transformer (∼15M weights), however, is still small by self-supervised transformer standards. We hypothesize that the representational capacity of larger transformers is simply not necessary for the relatively-simple datasets that are available in the MALDI-TOF MS domain. Because of this, we argue that more efforts to collect and publish data may be the most important factor for MALDI-TOF-based machine learning research to continue to flourish. Using more complex data, it is possible to formulate more complicated learning tasks, such as the recommender system in [15], that benefit more from the increased representational capacity of Maldi Transformer. As spectral data is routinely generated within hundreds, if not thousands, of hospitals, we envision collaboration efforts with healthcare (as in Weis et al. [10]) to play a crucial role in continued data collection and publishing.

Deep learning models have been hailed as feature extractors. For this class of models, it is typically argued that it matters less how features are presented to the model, as they can compose the relevant features themselves in their hidden representations. We would caution against this perspective, and state that how inputs are presented to a deep learning model could have far-reaching impacts on what the model can learn. Taking language as an example, character-level language models traditionally underperform the same models trained on (sub)word-level. In this work, MALDI-TOF mass spectra are presented to the model by their peaks. Because of this, any preprocessing steps and algorithms to determine peaks play crucial roles in the final performance of the model. Here, we present a small ablation study (Fig. E.11), but we identify further experimentation with peak detection and preprocessing algorithms as a promising future research direction.

Our novel peak shuffling pre-training paradigm outperforms MLM techniques that have been ported from language and have been proposed in the metabolomics MS data domain [30]. MLM pre-training introduces noise in the spectrum by obfuscation. We hypothesize that MLM pre-training does not match the performance of peak shuffling due to noisy optimization. Consider, for example, a masked (m/z) language model. Given an intensity and a masked (m/z) of a peak, multiple possible (m/z) locations might exhibit a peak with said intensity. As the ground truth pre-masking (m/z) value only represents one of multiple possible ground truths, the model is penalized for all but one of many correct answers. For this reason, we argue that MLM pre-training is ill-suited to the MS domain. It is expected that our proposed pre-training task design could be useful in other mass spectral domains. For this reason, we hope that our ablations (Section 3.2) and discussion in Appendix C serve as useful guidelines for adapting this work to other mass spectral data modalities, generated with MS instruments using alternative ionization and separation methods. One potential limitation of our method is that it only computes a loss for a certain percentage of peaks per spectrum, similarly to MLM. A line of further research could, therefore, be to adapt autoregressive pre-training to the spectral domain using domain knowledge.

As the pre-trained model can be interpreted as learning peak co-occurrences, this property is examined in Section 3.3. There, it is shown that Maldi Transformer can be used to detect noisy peaks. The predicted absence of noisy peaks is found to correlate with higher predictive performance. These insights go beyond those offered by off-the-shelf machine learning techniques. Paired with its state-of-the-art (or competitive) performance results, it can be concluded that Maldi Transformer enhances what biological practitioners get out of their MALDI-TOF MS data.

CRediT authorship contribution statement

Gaetan De Waele: Writing – review & editing, Writing – original draft, Software, Methodology, Investigation, Funding acquisition, Formal analysis, Conceptualization. **Gerben Menschaert:** Writing – review & editing, Supervision, Project administration, Conceptualization. **Peter Vandamme:** Writing – review & editing, Data curation, Conceptualization. **Willem Waegeman:** Writing – review & editing, Supervision, Project administration, Funding acquisition, Conceptualization.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Acknowledgments

This work was supported by Research Foundation-Flanders (FWO) [PhD Fellowship fundamental research grant 1153024N to G.D.W.]. W.W. also received funding from the Flemish Government under the "Onderzoeksprogramma Artificiële Intelligentie (AI) Vlaanderen" Programme

Appendix A. DRIAMS species processing

DRIAMS contains species labels for many spectra. These species labels are derived from the species identification pipelines included with the MALDI-TOF MS machines. As such, some preprocessing steps are taken to make the labels as presentable as possible to an ML model.

Firstly, labels in DRIAMS containing the string `''not reliable identification''` are integrally deleted. Second, many spectra are labeled as `''MIX!species''`, indicating that the spectrum potentially contains an impure mixture of species. As such, species labels for these spectra are also not used. Additionally, species labels that occur fewer than five times in the training set are removed. Finally, species labels occurring only in the validation or test set, but not in the training set, are similarly removed. After processing, DRIAMS contains 469 different species labels. Note that the previous steps involve removing of labels, not spectra themselves. Corresponding spectra are still kept in the dataset, albeit without a species label.

Appendix B. On preprocessing and persistence transformation

While Weis et al. [13] argue that persistence transformation nullifies the need for the parameter-heavy chain of preprocessing steps, we advocate for the opposite. To support this stance, a simple exploratory visualization is made.

In Fig. B.5, all *Bacillus anthracis*[8] spectra in the RKI training set are put through two preprocessing chains. The first only does (1) trimming to the 2000–20,000 Da range, (2) intensity calibration so that the total intensity sums to 1, and (3) persistence transformation, and keeps the 200 highest peaks. The second preprocessing chain performs all those steps preceded by variance stabilization, smoothing, and background removal, as in Section 2.1. Then, across all preprocessed spectra, the occurrence of detected peaks in bins of 3 Da is counted and plotted in Fig. B.5. The resulting profile can be considered a summary profile of all detected peaks in *Bacillus anthracis* spectra. The detected peaks with extra preprocessing result in a cleaner profile, with peaks more concentrated in regions that clearly correspond to biologically-informative signal (especially notable when inspecting the left tail of the spectra). In contrast, without prior preprocessing, detected peaks are seemingly more-randomly distributed, hinting that noise in the spectrum negatively affects peak detection.

Appendix C. Design of a self-supervised pre-training task

A large part of this study concerns the design of a self-supervised pre-training task for MALDI-TOF MS data. This is motivated by the scarcity of data in this domain. Through self-supervised learning, a greater representational capacity can be obtained, maximally utilizing patterns in the data. The following paragraphs chronicle thoughts and experiments w.r.t. the design of the SSL task.

[8] *Bacillus anthracis* is chosen because it is the most occurring species in the RKI training dataset.

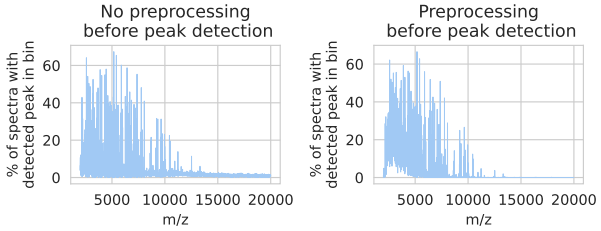

Fig. B.5. Visualization of summary profiles of *Bacillus anthracis* RKI training spectra obtained when doing peak detection with (R) or without (L) preprocessing first.

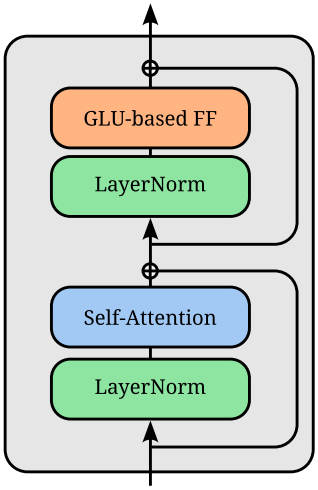

Fig. E.6. Utilized transformer encoder block. The network uses pre-LayerNorms and GeLU gated linear units (GLU) in the feedforward (FF) networks [37,38].

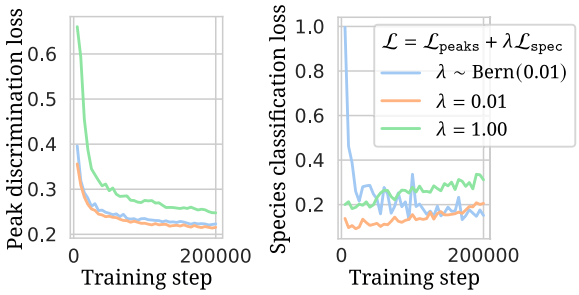

Fig. E.7. Pre-training dynamics using different strategies for combining the peak discrimination loss $\mathcal{L}_{\text{peaks}}$ and the species identification loss $\mathcal{L}_{\text{spec}}$. Loss on the validation set is shown. The figure is shown only for the first 200 000 pre-training steps using a medium-sized Maldi Transformer to illustrate. It is observed that if $\mathcal{L}_{\text{spec}}$ is applied at every step (using a weight of 0.01 or 1.00), this component starts overfitting well before the $\mathcal{L}_{\text{peaks}}$ component is converged (training is performed for 500 000 steps total). By not applying $\mathcal{L}_{\text{spec}}$ at every training step, the Adam optimizer momentum is dominated by $\mathcal{L}_{\text{peaks}}$. As $\mathcal{L}_{\text{spec}}$ is the "easier" task and more prone to overfitting, this regime benefits the final model.

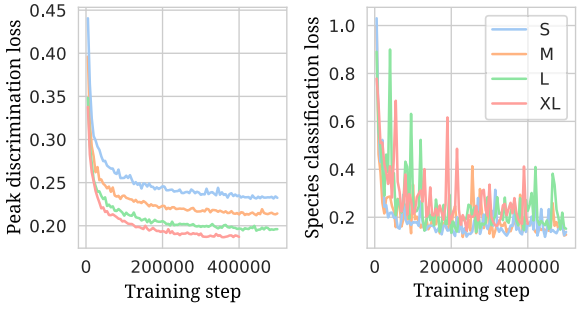

Fig. E.8. Pre-training validation loss curves for all Maldi Transformer model sizes.

MALDI-TOF MS spectra can be represented as sets of peaks. As each peak is characterized by an intensity and an (m/z) value, a natural parallel is drawn with language data. Instead of a sequence of words, a sequence of peaks is processed. A crucial difference is that, for language transformers, positional indices for each text "token" are set to the integer range numbering 0 to $n-1$, with n the number of tokens in the input sequence. With MALDI-TOF MS spectra, positional indices (i.e. (m/z) values) are irregularly spaced and real-valued. Additionally, whereas words have a categorical identity in text, MALDI-TOF MS intensity is also real-valued. These two factors have to be taken into consideration when porting self-supervised learning techniques from one modality to the mass spectral domain.

In language transformer pre-training, perhaps the two most-classically quoted techniques are masked language modeling (MLM) [29] and autoregressive modeling (AR) [39]. The GPT series of models best exemplifies the success of the latter category [40]. For MALDI-TOF MS, however, it is unclear how to autoregressively model a set of peaks. Autoregressive models imply a certain ordering of data, whereas this perspective is ill-fitting for the set-valued peaks input. For example, how does one train to generate the "next" peak given the previous peaks, if it is unclear how to define what the "next" peak is? For example, does one choose to order peaks by their height, or by their (m/z) value? If the model predicts the second-next peak instead of the next, is that necessarily wrong? If not, how to efficiently construct the loss to take this into account?

Instead of answering the issues with autoregressively modeling a set-valued input, we may consider the alternative strategy: MLM, introduced by BERT [29]. For example, instead of masking out words, intensities can be masked, and a model can be trained to recover the intensities with the use of the mean-square error loss. This brings us to first ablated pre-training technique in Section 3.2. The training for this strategy is performed identical to the final strategy outlined in Section 2.2. The only difference being that the peak discrimination loss $\mathcal{L}_{\text{peaks}}$ is exchanged for the mean-square error loss on masked peaks. Peaks to train on are similarly selected with 15% probability. Of those 15%, 80% are assigned masked intensities and 20% are left unchanged.

After fitting a regression MLM on intensities with limited success, a logical next step is to design a self-supervised classification task. A first choice may be to bin intensity values, and instead of regressing the masked intensities, classifying the intensity bin of the masked-out peaks. Here, we bin the intensity into 10,000 possible bins, and pre-train using a 10,000-way classification task. A different route is to mask out the (m/z) values of the peaks instead, and similarly train a MLM to classify the (m/z) bin of the masked-out peaks. This approach has been previously described in Bushuiev et al. [30]. Here, we bin the 2000–20,000Da (m/z)-range into 1Da bins, and pre-train using a 18,000-way classification task. Both methods are identically trained as the Intensity Regression MLM, described in the previous paragraph, but then using multi-class cross-entropy loss functions.

While MLMs on binned values already improve downstream performance results, some conceptual issues arise w.r.t. their application in the mass spectral data domain. Firstly, the masking procedure introduces a [MASK] token, which is only encountered during pre-training and fine-tuning. Hence, supervised fine-tuning is hampered due to a data distribution shift. Secondly, given a masked value of either intensity or (m/z), multiple possible answers may be feasible. For example, consider a masked (m/z) LM trained to predict the (m/z) of a peak given its intensity. Multiple locations for this peak may be feasible, but the model will penalize all but one. Hence, the plurality of ground-truth is not well-represented in the loss, and model pre-training is hampered. In order to overcome these issues, alternative pre-training techniques, taking inspiration from contrastive learning [19], are proposed. The following paragraphs describe the exact procedure resulting in the negative peak sampler model in Section 3.2.

Akin to contrastive learning, a per-peak classification task can ask a model whether a peak belongs to the rest of the spectrum or not.

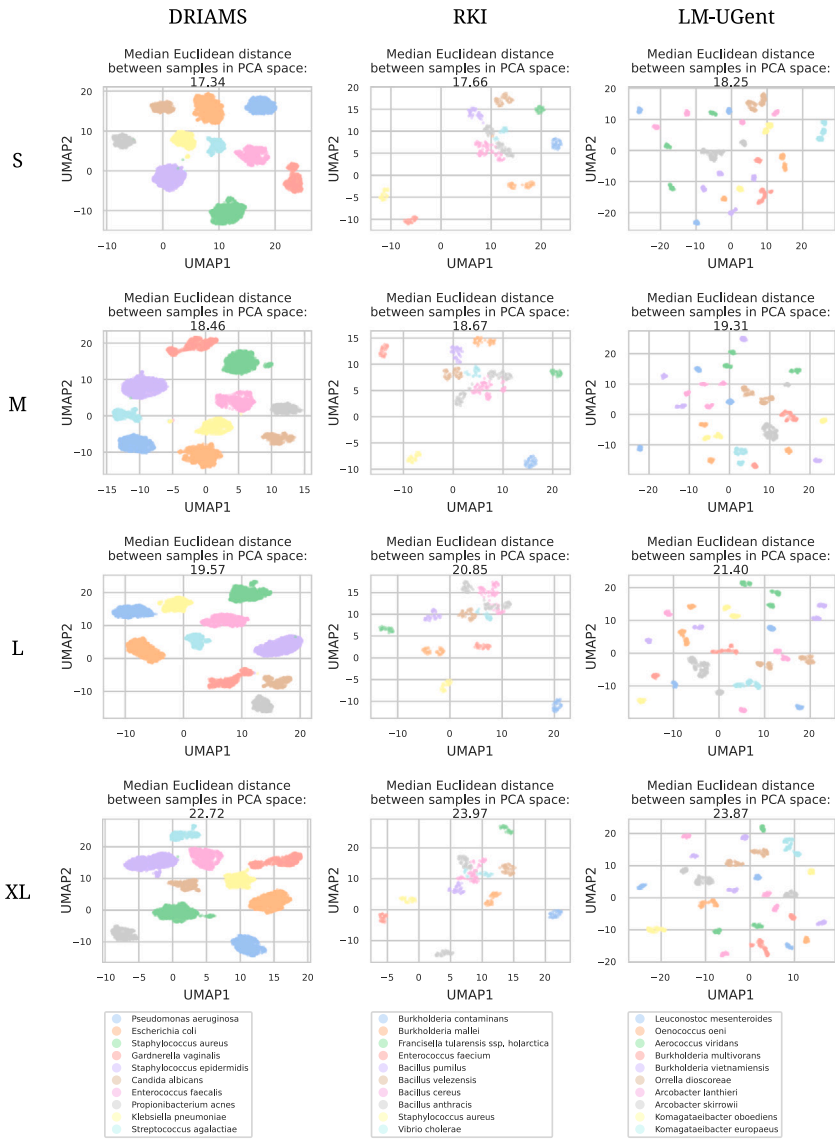

Fig. E.9. UMAP Embeddings of all validation spectra of all used datasets (columns), for every pre-trained (before fine-tuning) Maldi Transformer model size (rows). Only the 10 most occurring species in each dataset are used (legend). UMAP embeddings are obtained using $\boldsymbol{p}_{[\mathrm{CLS}]}$, standard normalized, then PCA transformed to 50 dimensions, then embedded in 2-dimensional space using UMAP. On top of every feature is the median euclidean distance between samples in the PCA space, before UMAP. It can be observed that, despite normalization and transforming to equal-dimensional spaces with PCA, the larger the pre-trained model, the more separated embeddings become. Notwithstanding, all models qualitatively find the same clustering.

Just as in contrastive learning, this strategy requires to sample negative peaks to deliver negative samples. One way to generate negative peaks is to randomly generate them from some estimated distribution of peaks. Here, we estimate the distribution of peaks in two steps. First, all peak locations in the training dataset are collected: $P_{\text{train}} = \left\{ \left(\frac{m}{z}\right)_j \middle| \left(\frac{m}{z}\right)_j \in S_i \right\}_{i=1}^{n}$, and a probability mass function over discrete bins is calculated:

$$\Pr(b_i) = \frac{\left| \{ p_j | p_j \in P_{\text{train}} \wedge p_j \in b_i \} \right|}{\left| \{ p_j | p_j \in P_{\text{train}} \} \right|} \tag{1}$$

with bins chosen by 1 Da intervals: $b_i \in]i, i+1]$, for $i \in \{2000, 2001, \ldots, 19\,999\}$. In other words, a probability mass function over 1 Da bins is created by counting how many times peaks fall into each bin in the training data. Next, within each bin, quantiles of the found intensities therein are calculated: $Q_{b_i} = \{q_{0.00}, q_{0.01}, \ldots, q_{0.99}, q_{1.00}\}$. To sample a negative peak, an (m/z) value is first sampled by uniformly sampling a location within a sampled bin: $(m/z) \sim \Pr(b_i) + \mathrm{Unif}[0, 1]$. After, an intensity is drawn by uniformly sampling within a uniformly sampled interquantile range: $I \sim \mathrm{Unif}[q_j, q_{j+1} \mid q \in Q_{b_i}]$, with $j \sim \mathrm{Unif}\{0.00, 0.01, \ldots, 0.99\}$. The training for this strategy is performed identical to the final strategy outlined in Section 2.2. The only difference being that, instead of shuffling peaks around to generate negatives, negative peaks are now sampled. In every training step, 15% of the peaks are selected for training, half of which are exchanged for sampled negative ones.

As discussed in Section 3.2, generating negative peaks is not ideal for forcing the model to reason over peak co-occurrences. The model is allowed to learn any misrepresentation in negatives as a shortcut. A way to circumvent the issues with generating negative peaks, is to use

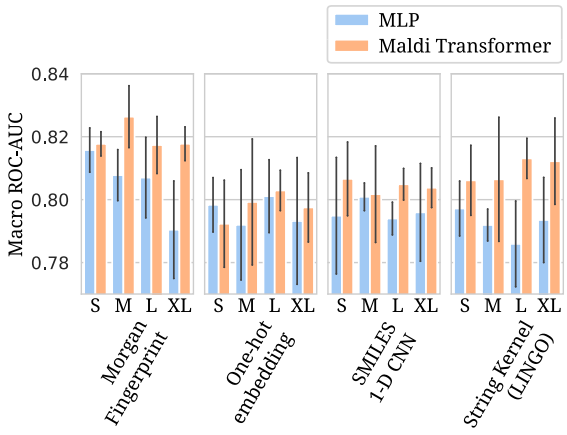

Fig. E.10. Barplots of AMR prediction on DRIAMS, evaluated by Macro (drug) ROC-AUC Error-bars indicate standard deviation over five independent runs. Neural network model sizes range from S to XL. Details per model size for Maldi Transformer and MLP models are listed in Tables 2 and D.3, respectively.

real ones. One way to present real peaks as negatives, is to take them from other spectra. This is where we land on the final self-supervised strategy that we test, and ultimately propose in our main text (see Section 2.2, Fig. 1, and Algorithm 1).

Algorithm 1: Maldi Transformer peak discrimination pre-training.

Input: $\boldsymbol{X} \in \mathbb{R}^{b\times m\times d}$, $\boldsymbol{y}^{\text{spec}} \in \mathbb{R}^{b}$

b = batch size, m = number of peaks, d = hidden dim

$\boldsymbol{X}$ = encoded batch of spectra, $\boldsymbol{y}^{\text{spec}}$ = species labels

$\boldsymbol{y}^{\text{spec}}$ contains int labels and NaN if unlabeled

Let $\boldsymbol{U} \in \{0,1\}^{b\times m}$, with $U_{ij} \sim \text{Bern}(0.15)$ # Sample 15% of peaks

Let $\boldsymbol{P} \in \{0,1\}^{b\times m}$, with $P_{ij} \sim \text{Bern}(0.50)$ # Sample 50% of peaks

$\boldsymbol{p} = [(i,j) : U_{ij} = 1 \wedge P_{ij} = 1]$ # Positive label indices

$\boldsymbol{n} = [(i,j) : U_{ij} = 1 \wedge P_{ij} = 0]$ # Negative label indices

$\boldsymbol{X}^{\text{shuff}} = \boldsymbol{X}[\texttt{permute}(\boldsymbol{n})]$ # Shuffle neg peaks

$\boldsymbol{P}_{j\in\{1,\dots,m\}}, \boldsymbol{P}_{[\text{CLS}]} = \texttt{MaldiTransformer}(\boldsymbol{X}^{\text{shuff}})$

$\hat{\boldsymbol{Y}}^{\text{peak}} = \sigma(\boldsymbol{w}_p^{\intercal} \cdot \boldsymbol{P}_{j\in\{1,\dots,m\}})$ # Projection of peak embeddings

$\hat{\boldsymbol{Y}}^{\text{spec}} = \text{Softmax}(\boldsymbol{W}_s^{\intercal} \cdot \boldsymbol{P}_{[\text{CLS}]})$ # Projection of spec embed

$\mathcal{L}_{\text{peaks}} = \frac{1}{|[(i,j):U_{ij}=1]|} \sum_{U_{ij}=1}^{(i,j)} \Big(-\mathbb{I}\left(X_{ij} = X_{ij}^{\text{shuff}}\right) \log\left(\hat{Y}_{ij}^{\text{peak}}\right) - \mathbb{I}\left(X_{ij} \neq X_{ij}^{\text{shuff}}\right) - \log\left(1 - \hat{Y}_{ij}^{\text{peak}}\right) \Big)$ # Binary CE on peaks

$\mathcal{L}_{\text{spec}} = \frac{1}{b} \sum_{i}^{b} -\log \hat{Y}_{i,c}^{\text{spec}}$ # with $c = y_i^{\text{spec}}$ the class index

$\mathcal{L} = \mathcal{L}_{\text{peaks}} + \lambda \cdot \mathcal{L}_{\text{spec}}$, with $\lambda \sim \text{Bern}(0.01)$

Output: $\mathcal{L}$ # Final loss

Appendix D. Downstream tasks and baselines

For all three downstream tasks, the pre-trained model is plugged in at initialization and all weights are fine-tuned (i.e. no weight freezing). A task-specific linear output head $\boldsymbol{W}_{out}$ projecting the spectrum embedding $\boldsymbol{p}_{[\text{CLS}]}$ to the desired output space is trained from scratch. All downstream models are similarly trained with the Adam optimizer with a batch size of 128 and dropout of 0.2. A linear learning rate warm-up over the first 250 steps is applied, after which the rate is kept constant.

For AMR prediction, training of Maldi Transformer-based recommenders is performed identical to the MLP-based baselines in [15]. Briefly explained, for every combination of spectrum embedder (four sizes: S, M, L, and XL) and drug embedder (four types), six different learning rates ({1e–5, 5e–5, 1e–4, 5e–4, 1e–3, 5e–3}) are tested. For all these different combinations, five models are trained (using different random seeds for model initialization and batching of data). For every spectrum and drug embedder combination, only results from the best learning rate are presented; that is, the learning rate resulting in the best average validation ROC-AUC for that combination. The validation set is checked every tenth of an epoch. Models are trained for a maximum of 50 epochs, and their training is halted early when validation ROC-AUC has not improved for 10 validation set checks. The checkpoint of the best performing model (in terms of validation ROC-AUC) is used as the final model. The baselines for the AMR prediction task are the models described in De Waele et al. [15], which describes the recommender model structure in greater detail.

Table D.3
All tested model sizes for the MLP baseline. Hidden sizes represent the evolution of the hidden state dimensionality as it goes through the model, with every hyphen defining one fully connected layer. n represents the number of output nodes. n equals 64, 270, and 1088 for DRIAMS AMR prediction, species identification on RKI, and LM-UGent, respectively.

Size	# Weights	Hidden sizes
S	1.58M	6000-256-128-n
M	3.25M	6000-512-256-128-n
L	6.85M	6000-1024-512-256-128-n
XL	15.09M	6000-2048-1024-512-256-128-n

The fine-tuning of Maldi Transformer for species identification is performed in a similar way. The differences consist of: (1) $\boldsymbol{W}_{out}$ returning 270, or 1088, for the RKI and LM-UGent dataset, respectively, (2) five different learning rates are tested: ({1e–5, 5e–5, 1e–4, 5e–4, 1e–3}), and (3) validation species-level accuracy is tracked to halt training early (for a maximum of 250 epochs, and it is only checked once per epoch). As species identification is a multi-class classification task, models are then optimized using a softmax operation, combined with the cross-entropy loss. Species identification is compared to MLP baselines, Logistic Regression, Random Forest, and k-nearest neighbors (k-NN) models. All of these baselines are trained on preprocessed and binned spectra (see Section 2.1).

The MLP baselines are identical in construction to the spectrum embedders in De Waele et al. [15], but with n-dimensional outputs instead of 64, with n the number of classes (see Table D.3). MLP baselines are trained using the same strategy as with Maldi Transformer downstream fine-tuning. That is, for all model sizes, five different learning rates ({1e–5, 5e–5, 1e–4, 5e–4, 1e–3}) are trained five-fold. Model results from the best learning rate (in terms of validation species-level accuracy) are presented. Model training halts before its maximum of 250 epochs if validation species-level accuracy has not increased in 10 epochs, and the model with the best validation accuracy is saved.

For non-neural baseline classifiers (Random Forest, Logistic Regression, and k-NN), a grid-search is performed to find optimal hyperparameters. Given the non-stochastic nature of their implementations, only one model is trained after hyperparameter tuning and, hence, only one test performance is reported. The parameter grid for Random Forest consists of {`max_depth = [25,50,75,100]`, `min_samples_split = [2,5,10]`, `max_features = [10,25, 50,100]`}. All random forests are trained with 200 trees. For Logistic Regression: {`standardscaling = [True,False]`, `L2_norm = [1e3,1e2,10,1,0.1,1e-2,1e-3]`}. And for k-NN: {`standard scaling = [True,False]`, `n_neighbors = [1,2, 3,4,5,6, 7,9,10,25]`}.

Appendix E. Figures and tables supporting the methods and results sections

See Figs. E.6–E.15.

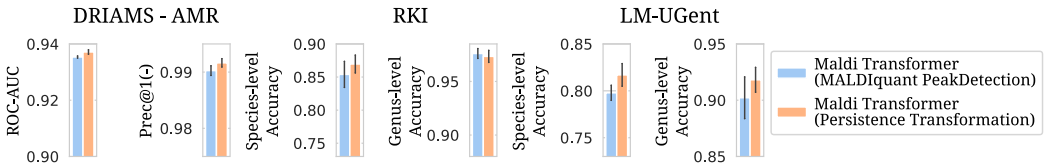

Fig. E.11. Barplots of all peak detection ablation results Only results for Medium-sized transformers are shown. The final model as discussed in Section 2.2 is labeled as Maldi Transformer (Persistence Transformation). The results for MALDIquant Peakdetection are obtained by pre-training and fine-tuning Maldi Transformer in exactly the same way, but with peaks detected using MALDIquant's methodology (reproduced in Python). Error-bars indicate standard deviation over five independent runs. For DRIAMS AMR prediction, only results using a Morgan Fingerprint drug embedder are shown. For the LM-UGent dataset, results are shown without the domain adaptation step, as this step was also not performed for ablation models.

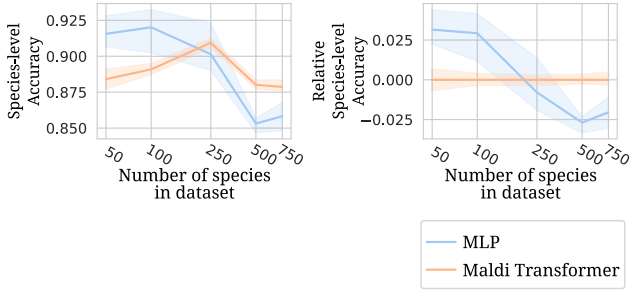

Fig. E.12. Lineplots of species-level accuracy on LM-UGent when trained and evaluated on a decreasing number of species. Species are filtered out on training data occurrency, always keeping the n-most-occurring species in the data. Errorband indicate standard deviation over three independent runs. The right hand plot shows the species-accuracy relative to the one obtained by Maldi Transformer.

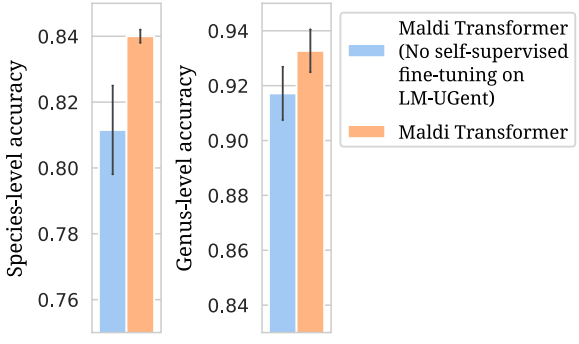

Fig. E.13. LM-UGent performance improves when, prior to supervised fine-tuning, the pre-trained model is first fine-tuned using the self-supervised training task. In the main text, we refer to this step as the domain adaptation step.

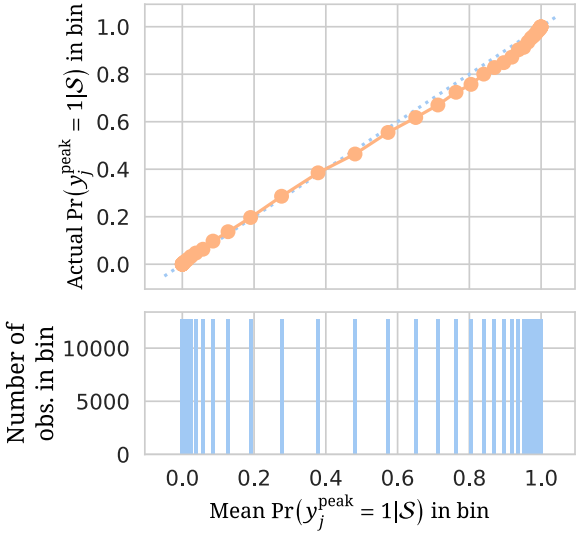

Fig. E.14. Calibration curve for pre-training peak discrimination using the medium-sized Maldi Transformer. On the DRIAMS test-set, the usual peak shuffling and peak discrimination is performed (see Section 2.2). Predictions are then sorted and split into equal frequency bins (on the x-axis). Within those bins, the average predicted value is plotted against the actual fraction of positive true labels in that bin. A calibrated model requires predictions to be interpretable in a frequentist manner, i.e. a sample with an output probability of 80% is expected to be positive 80% of the times. Hence, the aforementioned plot is expected to follow the diagonal. The plot shows that the model is reasonably-well calibrated, with slight overconfidence.

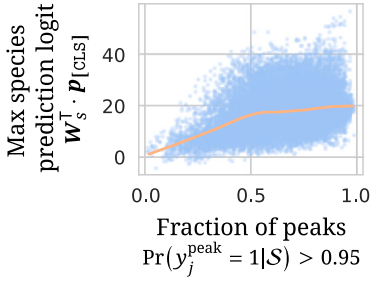

Fig. E.15. Max species prediction logit $\boldsymbol{W}_s^{\intercal} \cdot \boldsymbol{p}_{[\text{CLS}]}$ in function of number of confidently "belonging" peaks for a spectrum for DRIAMS test set spectra using the medium-sized Maldi Transformer. A higher max logit for the species prediction task corresponds to a higher max output probability post-softmax, and, hence, higher confidence. It is displayed that the model is more confident in its prediction for spectra with a higher number of confidently belonging peaks.

References

[1] O. Dauwalder, T. Cecchini, J.P. Rasigade, F. Vandenesch, Matrix assisted laser desorption ionisation/time of flight (MALDI/TOF) mass spectrometry is not done revolutionizing clinical microbiology diagnostic, Clin. Microbiol. Infect. 29 (2) (2023) 127–129.

[2] C.V. Weis, C.R. Jutzeler, K. Borgwardt, Machine learning for microbial identification and antimicrobial susceptibility testing on MALDI-TOF mass spectra: a systematic review, Clin. Microbiol. Infect. 26 (10) (2020) 1310–1317.

[3] P. Seng, M. Drancourt, F. Gouriet, B. La Scola, P.-E. Fournier, J.M. Rolain, D. Raoult, Ongoing revolution in bacteriology: routine identification of bacteria by matrix-assisted laser desorption ionization time-of-flight mass spectrometry, Clin. Infect. Dis. 49 (4) (2009) 543–551.

[4] A. Bizzini, K. Jaton, D. Romo, J. Bille, G. Prod'hom, G. Greub, Matrix-assisted laser desorption ionization–time of flight mass spectrometry as an alternative to 16s rRNA gene sequencing for identification of difficult-to-identify bacterial strains, J. Clin. Microbiol. 49 (2) (2011) 693–696.

[5] A. Van Belkum, M. Welker, M. Erhard, S. Chatellier, Biomedical mass spectrometry in today's and tomorrow's clinical microbiology laboratories, J. Clin. Microbiol. 50 (5) (2012) 1513–1517.

[6] W. Florio, A. Tavanti, S. Barnini, E. Ghelardi, A. Lupetti, Recent advances and ongoing challenges in the diagnosis of microbial infections by MALDI-TOF mass spectrometry, Front. Microbiol. 9 (2018) 1097.

[7] Y. Cao, L. Wang, P. Ma, W. Fan, B. Gu, S. Ju, Accuracy of matrix-assisted laser desorption ionization–time of flight mass spectrometry for identification of mycobacteria: a systematic review and meta-analysis, Sci. Rep. 8 (1) (2018) 1–9.

[8] G. Vrioni, C. Tsiamis, G. Oikonomidis, K. Theodoridou, V. Kapsimali, A. Tsakris, MALDI-TOF mass spectrometry technology for detecting biomarkers of antimicrobial resistance: current achievements and future perspectives, Ann. Transl. Med. 6 (12) (2018).

[9] J.M. Hettick, M.L. Kashon, J.E. Slaven, Y. Ma, J.P. Simpson, P.D. Siegel, G.N. Mazurek, D.N. Weissman, Discrimination of intact mycobacteria at the strain level: a combined MALDI-TOF MS and biostatistical analysis, Proteomics 6 (24) (2006) 6416–6425.

[10] C. Weis, A. Cuénod, B. Rieck, O. Dubuis, S. Graf, C. Lang, M. Oberle, M. Brackmann, K.K. Søgaard, M. Osthoff, et al., Direct antimicrobial resistance prediction from clinical MALDI-TOF mass spectra using machine learning, Nature Med. 28 (1) (2022) 164–174.

[11] J. Gagnaire, O. Dauwalder, S. Boisset, D. Khau, A.-M. Freydiere, F. Ader, M. Bes, G. Lina, A. Tristan, M.-E. Reverdy, et al., Detection of Staphylococcus aureus delta-toxin production by whole-cell MALDI-TOF mass spectrometry, PLoS One 7 (7) (2012) e40660.

[12] K. Vervier, P. Mahé, J.-B. Veyrieras, J.-P. Vert, Benchmark of structured machine learning methods for microbial identification from mass-spectrometry data, 2015, arXiv preprint arXiv:1506.07251.

[13] C. Weis, M. Horn, B. Rieck, A. Cuénod, A. Egli, K. Borgwardt, Topological and kernel-based microbial phenotype prediction from MALDI-TOF mass spectra, Bioinformatics 36 (Supplement_1) (2020) i30–i38.

[14] T. Mortier, A.D. Wieme, P. Vandamme, W. Waegeman, Bacterial species identification using MALDI-TOF mass spectrometry and machine learning techniques: A large-scale benchmarking study, Comput. Struct. Biotechnol. J. 19 (2021) 6157–6168.

[15] G. De Waele, G. Menschaert, W. Waegeman, An antimicrobial drug recommender system using MALDI-TOF MS and dual-branch neural networks, 2023, http://dx.doi.org/10.1101/2023.09.28.559916.

[16] N.K. Tran, T. Howard, R. Walsh, J. Pepper, J. Loegering, B. Phinney, M.R. Salemi, H.H. Rashidi, Novel application of automated machine learning with MALDI-TOF-MS for rapid high-throughput screening of COVID-19: A proof of concept, Sci. Rep. 11 (1) (2021) 8219.

[17] V. Ryzhov, C. Fenselau, Characterization of the protein subset desorbed by MALDI from whole bacterial cells, Anal. Chem. 73 (4) (2001) 746–750.

[18] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A.N. Gomez, Ł. Kaiser, I. Polosukhin, Attention is all you need, Adv. Neural Inf. Process. Syst. 30 (2017).

[19] X. Liu, F. Zhang, Z. Hou, L. Mian, Z. Wang, J. Zhang, J. Tang, Self-supervised learning: Generative or contrastive, IEEE Trans. Knowl. Data Eng. 35 (1) (2021) 857–876.

[20] R. Balestriero, M. Ibrahim, V. Sobal, A. Morcos, S. Shekhar, T. Goldstein, F. Bordes, A. Bardes, G. Mialon, Y. Tian, et al., A cookbook of self-supervised learning, 2023, arXiv preprint arXiv:2304.12210.

[21] J. Clauwaert, W. Waegeman, Novel transformer networks for improved sequence labeling in genomics, IEEE/ACM Trans. Comput. Biol. Bioinform. 19 (1) (2020) 97–106.

[22] J. Jumper, R. Evans, A. Pritzel, T. Green, M. Figurnov, O. Ronneberger, K. Tunyasuvunakool, R. Bates, A. Žídek, A. Potapenko, et al., Highly accurate protein structure prediction with AlphaFold, Nature 596 (7873) (2021) 583–589.

[23] Ž. Avsec, V. Agarwal, D. Visentin, J.R. Ledsam, A. Grabska-Barwinska, K.R. Taylor, Y. Assael, J. Jumper, P. Kohli, D.R. Kelley, Effective gene expression prediction from sequence by integrating long-range interactions, Nature Methods 18 (10) (2021) 1196–1203.

[24] A. Elnaggar, M. Heinzinger, C. Dallago, G. Rehawi, Y. Wang, L. Jones, T. Gibbs, T. Feher, C. Angerer, M. Steinegger, et al., Prottrans: Toward understanding the language of life through self-supervised learning, IEEE Trans. Pattern Anal. Mach. Intell. 44 (10) (2021) 7112–7127.

[25] M. Yilmaz, W. Fondrie, W. Bittremieux, S. Oh, W.S. Noble, De novo mass spectrometry peptide sequencing with a transformer model, in: International Conference on Machine Learning, PMLR, 2022, pp. 25514–25522.

[26] T. Yang, T. Ling, B. Sun, Z. Liang, F. Xu, X. Huang, L. Xie, Y. He, L. Li, F. He, et al., Introducing π-HelixNovo for practical large-scale de novo peptide sequencing, Brief. Bioinform. 25 (2) (2024) bbae021.

[27] S. Goldman, J. Wohlwend, M. Stražar, G. Haroush, R.J. Xavier, C.W. Coley, Annotating metabolite mass spectra with domain-inspired chemical formula transformers, Nat. Mach. Intell. 5 (9) (2023) 965–979.

[28] T. Butler, A. Frandsen, R. Lightheart, B. Bargh, T. Kerby, K. West, J. Davison, J. Taylor, C. Krettler, T. Bollerman, et al., MS2Mol: A transformer model for illuminating dark chemical space from mass spectra, ChemRxiv (2023) http://dx.doi.org/10.26434/chemrxiv-2023-vsmpx-v4.

[29] J. Devlin, M.-W. Chang, K. Lee, K. Toutanova, Bert: Pre-training of deep bidirectional transformers for language understanding, 2018, arXiv preprint arXiv:1810.04805.

[30] R. Bushuiev, A. Bushuiev, R. Samusevich, C. Brungs, J. Sivic, T. Pluskal, Emergence of molecular structures from repository-scale self-supervised learning on tandem mass spectra, ChemRxiv (2024) http://dx.doi.org/10.26434/chemrxiv-2023-kss3r-v2.

[31] P. Lasch, M. Stämmler, A. Schneider, Version 4 (20230306) of the MALDI-ToF mass spectrometry database for identification and classification of highly pathogenic microorganisms from the Robert Koch-Institute (RKI), 2023, http://dx.doi.org/10.5281/zenodo.7702375.

[32] S. Gibb, K. Strimmer, MALDIquant: a versatile R package for the analysis of mass spectrometry data, Bioinformatics 28 (17) (2012) 2270–2271.

[33] I. Beltagy, M.E. Peters, A. Cohan, Longformer: The long-document transformer, 2020, arXiv preprint arXiv:2004.05150.

[34] A. Dosovitskiy, L. Beyer, A. Kolesnikov, D. Weissenborn, X. Zhai, T. Unterthiner, M. Dehghani, M. Minderer, G. Heigold, S. Gelly, et al., An image is worth 16x16 words: Transformers for image recognition at scale, 2020, arXiv preprint arXiv:2010.11929.

[35] S. Narang, H.W. Chung, Y. Tay, W. Fedus, T. Fevry, M. Matena, K. Malkan, N. Fiedel, N. Shazeer, Z. Lan, et al., Do transformer modifications transfer across implementations and applications?, 2021, arXiv preprint arXiv:2102.11972.

[36] D.P. Kingma, J. Ba, Adam: A method for stochastic optimization, 2014, arXiv preprint arXiv:1412.6980.

[37] J.L. Ba, J.R. Kiros, G.E. Hinton, Layer normalization, 2016, arXiv preprint arXiv:1607.06450.

[38] N. Shazeer, Glu variants improve transformer, 2020, arXiv preprint arXiv:2002.05202.

[39] A. Radford, K. Narasimhan, T. Salimans, I. Sutskever, et al., Improving language understanding by generative pre-training, 2018.

[40] T. Brown, B. Mann, N. Ryder, M. Subbiah, J.D. Kaplan, P. Dhariwal, A. Neelakantan, P. Shyam, G. Sastry, A. Askell, et al., Language models are few-shot learners, Adv. Neural Inf. Process. Syst. 33 (2020) 1877–1901.