Machine Learning for Direct Antimicrobial Resistance Prediction from clinical MALDI-TOF Spectra

by
Katharina Hagedorn

Bachelor's Thesis in Informatics
School of Computation, Information and Technology - Informatics
at
Technical University of Munich

Machine Learning for Direct Antimicrobial Resistance Prediction from clinical MALDI-TOF Spectra

Maschinelles Lernen für direkte antimikrobielle Resistenz Vorhersagen mit klinischen MALDI-TOF Massenspektren

Author:	Katharina Hagedorn
Supervisor:	Prof. Dr. Niki Kilbertus
Advisor:	Dr. Bastian Rieck
Submission Date:	May 15, 2024

Bachelor's Thesis in Informatics
School of Computation, Information and Technology - Informatics
at
Technical University of Munich

I confirm that this bachelor's thesis is my own work and I have documented all sources and material used.

Munich, May 15, 2024 Katharina Hagedorn

Abstract

The spread of antimicrobial resistance across the world poses one of the biggest threats to modern medicine. The severity of this phenomenon has sparked research on machine-learning approaches predicting the resistance attributes of microbe species. Implementing resistance classification into the clinical workflow necessitates fast predictions and, thus, fast retrieval of data representing the microbes. Matrix-assisted laser desorption/ionisation time-of-flight mass spectrometry provides mass spectra of microbe species within under 24 hours. Such data is successfully used to predict the resistance of microbes from one location. However, the challenge remains that microbe spectra differ depending on their location of retrieval, present classifiers are not able remedy these discrepancies in spectra and thus fail to consistently classify microbes using data from multiple locations; their generalisation ability is weak. Here, we exhibit model exploration and ablation of the deep convolutional neural networks LeNet5, AlexNet, DenseNet and Vgg16 to increase the generalisation performance and set the basis for future transfer learning model architectures. The classifiers' performances are being compared to previous contributions in terms of AUROC and AUPRC performance. We validate our models using ten different random train-test splits for 16 different train-test scenarios from the four domains DRIAMS-A–DRIAMS-D provided by the DRIAMS dataset. Our baseline model, a multilayer perceptron, sets the upper bound of AUROC performance with a mean AUROC value of 0.67, whereas the LeNet5 and AlexNet classifier reach scores of 0.66 and 0.62, respectively. The deep convolutional neural networks perform very similarly to classifiers from previous work using the DRIAMS-Dataset. This implies that the resistance mechanisms within the data cannot be identified by various models, suggesting that the data possibly does not provide this information.

Contents

List of Abbreviations

AMR	Antimicrobial Resistance
ANN	Artificial Neural Network
AUPRC	Area Under the Precision-Recall Curve
AUROC	Area Under the Receiver Operating Characteristic Curve
CEF	Ceftriaxone
CLSI	Clinical and Laboratory Standards Institute
CNN	Convolutional Neural Network
CPU	Central Processing Unit
CUDA	Compute Unified Device Architecture
Da	Dalton
DNN	Deep Neural Network
DRIAMS	Database of Resistance Information on Antimicrobials and MALDI-TOF Mass Spectra
DT	Decision Tree
GPU	Graphics Processing Unit
KNN	k-Nearest Neighbours
LightGBM	Light Gradient Boosting Machine
MALDI-TOF	Matrix Assisted Laser Desorption Ionization
MLP	Multi-Layer Perceptron
MS	Mass Spectrometry
RAM	Random-Access Memory
RF	Random Forest
SNN	Siamese Neural Network
SVM	Support Vector Machine
WHO	World Health Organization
W&B	Weights & Biases

1 Introduction

The WHO approximates that 1.43% of the world's population consumes antibiotics on any given day of the year; in other words, precisely 11.207.900 people undergo antibiotic treatment daily [5]. First and foremost, this fact highlights the remarkable progress made in modern medicine through the exploration of antibiotics. Still, it also foretells a picture of crisis when imagining a world without antibiotics.

In recent years, scientists have observed growing numbers of resistant microbes against various antibiotics, and they are currently speaking of the **Antimicrobial Resistance Crisis**. Amongst others, a critical variable in combating advancing resistance is fast and narrow antibiotic treatment, as opposed to broad antibiotic treatment. In this thesis, we roughly describe a workflow to realise faster antimicrobial resistance predictions in clinically relevant scenarios. Our focus lies on machine learning methods designed to predict the resistance values of specific species–antibiotic combinations. This application of machine learning has gained significant attention in recent years, especially when predicting on genome sequencing data of microbes. Yet, the acquisition of MALDI-TOF spectra of microbes is much faster than, for example, genomic sequencing; therefore, using MALDI-TOF spectral data enables more rapid predictions. Accordingly, we use and extend the prior publication "Direct antimicrobial resistance prediction from clinical MALDI-TOF mass spectra using machine learning"[38], which provides preprocessing of MALDI-TOF spectra as well as numerous machine learning approaches.

An ever-occurring challenge is that classifiers can predict very well on specific datasets and specific species–antibiotic combinations but fail to generalise to other unseen datasets. Hence, our main objective is to increase our classifier's generalisation performance on our chosen dataset (DRIAMS); for this reason, we primarily explore deep neural networks in the given context. More precisely, we begin by applying various deep convolutional neural network classifiers to our dataset. Subsequently, we analyse the corresponding results and model behaviour more thoroughly by conducting an ablation study. In general, we find that deep convolutional neural networks can compete with the existing traditional machine learning methods, like MLPs, for AMR prediction using the DRIAMS dataset. As we strongly depend on the need for low computational complexity, we identify a modified version of the LeNet5 model architecture as our most performant model through our ablation study, achieving a mean AUROC score of 0.66. In comparison, the MLP achieves an AUROC of 0.67. Nevertheless, we observe that the differences in model performance are marginal when applying different model architectures.

This thesis aims to demonstrate the potential of deep convolutional neural networks applied to AMR prediction tasks. An essential component of this task is the DRIAMS dataset, we detail the data characteristics by explaining the fundamental concepts AMR, MALDI-TOF Mass Spectrometry and binnning. To set the context, we give an overview of competing machine learning methods and give a detailed explanation of the publications we extend, as this is the basis for

our work. We reveal our data preparation steps using PyTorch and detail our model architectures and parameters. Finally, we exhibit the experiments that were conducted, as well as our classifiers' performances and ablation results.

2 Background

2.1 Antimicrobial Resistance

Microbes, meaning organisms like bacteria, viruses, parasites and fungi[17] have the ability to adapt to drugs over time as they undergo adaptive evolutionary changes[14]. Due to these changes, microbes can become unresponsive or less responsive to drugs they were initially sensitive to, namely to antimicrobials[14]. The result of this genomic change is known as the organism's direct antimicrobial resistance[17]. Antimicrobial resistance has been declared one of the gravest global **public health** and **development** threats by numerous organisations and researchers[23] such as the WHO[26].

The AMR crisis is of great extent because it jeopardises the gains of modern medicine: AMR not only affects the treatment of a manifold of diseases, but an increased occurrence of AMR also increases the risk for routine medical procedures such as surgery, caesarean sections and cancer chemotherapy possibly leading to grave illness or death[24][17]. To put the threat to **public health** into perspective, Murray et al.[23] estimate that AMR specifically attributed to bacteria was, directly and indirectly, responsible for about five million deaths in 2019 (solely direct: 1.270.000, total: 4.950.000) [23] and a report commissioned by the UK Prime minister from 2016 suggests that 10 million deaths globally will be attributable to AMR per year by 2050[24].

The threat to **development** is portrayed in a report by the World Bank[3]; they estimate that by 2050, the annual global gross domestic product will fall by 1.1% in the best case, by 3.8% in the worst case, relative to the base-case of no AMR. They also state that low-income countries will suffer more significant drops in economic growth and experience a greater increase in healthcare expenditures (low-income: 25%, middle income: 15%, high-income: 6%)[3].

Murray et al.[23] published an extensive study on the burden of AMR in 2022, showing that six pathogens are responsible for more than 250,000 deaths associated with AMR. These are the following bacteria sorted in the order of number of deaths descendingly: E. coli, Staphylococcu aureus, K. pneumoniae, S. pneumoniae, Acinetobacter baumannii, and Pseudomonas aeruginosa, together, they make up 929,000 of 1.27 million direct deaths and 3.57 million of 4.95 million total deaths globally in 2019. The deathly pathogens are distributed differently across the globe; in the high-income super-region, S. Aureus (constituting 26.1% of deaths attributable to AMR and 25.4% of deaths associated with AMR) and E. coli (constituting 23.4% of deaths attributable to AMR and 24.3% of deaths associated with AMR) contribute to half of the deaths attributable to AMR. In sub-Saharan Africa, per contra, S. pneumoniae (15.9% of the deaths attributable to AMR and 19.0% of the deaths associated with AMR) and K. pneumoniae (19.9% of the deaths attributable to AMR and 17.5% of the deaths associated with AMR). The leading pathogens make up a smaller proportion of the AMR burden. Consequently, we will focus on K. pneumoniae and E. coli in this thesis.

Factors affecting the spread of AMR are predominantly the "misuse and overuse of antibiotics"[14] in humans[14] as well as in animal and plant care[13][24]. In humans, treatments often rely on broad-spectrum antibiotics to quickly limit the infection-related risk to the patient[14], this unnecessary, indiscriminate broad antibiotic use indirectly facilitates antibiotic resistance[38][13]. As Caroline Weis et al.[38] point out, an important aspect in inhibiting AMR is the fast availability of resistance profiles of bacteria to be able to incorporate this knowledge into diagnosis and treatment. Fast availability (under 24 hours) would enable a narrow antibiotic use and accelerate infection prevention measures.[38]

2.2 MALDI-TOF Mass Spectrometry

The data we utilise in this work are MALDI-TOF spectra. These spectra are produced in a process called MALDI-TOF mass spectrometry, which characterises microorganisms, including bacteria, fungi, and viruses[7]. It creates highly specific microbe-spectra in only 24 hours[38]. Due to its low cost, reliability [42] and high throughput[40] MALDI-TOF mass spectrometry is an indispensable tool in various fields such as medical diagnostics, biodefense, environmental monitoring, and food quality control.[7, 32] In the following we want to further explain the terms Mass spectrometry, matrix-assisted laser desorption ionisation (MALDI) and time-of-flight (TOF):
Mass spectrometry creates mass spectra from sample molecules, this process can be divided into the following phases: ionisation, ion separation, detection[7]. There are multiple ways to achieve ionisation, a widely used method for microbial species identification is MALDI, a soft ionisation technique using laser impulses [7, 32]. The sample is mixed with the matrix on a conductive metal plate, resulting in crystallisation. This mixture is introduced to the mass spectrometer. The aforementioned laser, commonly a nitrogen laser, bombards the sample mixture with high energy laser pulses. This is a soft ionisation technique[34], as the matrix absorbs the energy from the laser and transmits it to the sample, while only minimally fragmenting the molecules. This ionises and desorbs the sample into the gas phase, resulting in predominantly singly charged sample ions[32, 34].

After Ionisation the mass spectrometer's electrostatic field accelerates the analytes through a metal flight tube until they reach the detector device which monitors the arriving ions.[7] As the ions have the same charge (z) but different masses, the acceleration leads to the ions seperating according to their mass (m) because "the time of flight is proportional to the square root of m/z"[7] .[7, 32] The resulting mass spectrum shows the ion signal as a function of the m/z–ratio, giving insights into the contents of the sample.[42] In spectra acquired by MALDI-TOF, spectrum spikes range between 2,000-20,000 Da.[7]

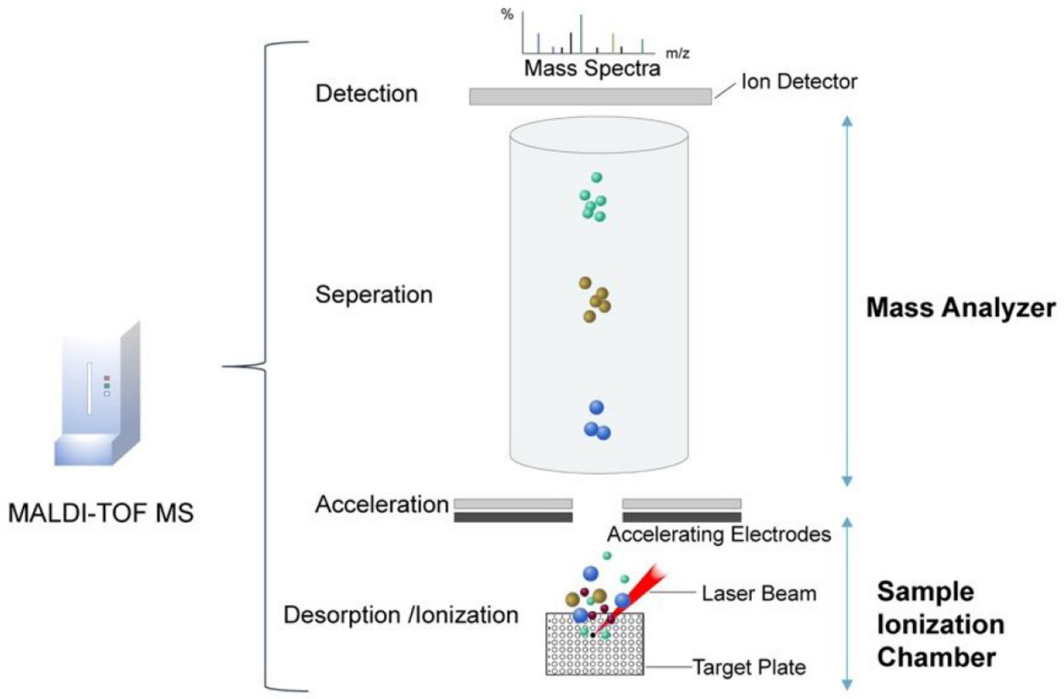


Figure 2.1: This figure is reproduced from Li et al. [18], it shows the process of MALDI-TOF Mass Spectrometry: A laser beam ionises the sample in the Sample Ionization Chamber. Electrodes then accelerate the ions, which leads to a separation within the Mass Analyzer. At the end of the Mass Analyzer, the Ion Detector generates mass spectra from the detected ions. [18]

2.3 DRIAMS Dataset

In the scope of this thesis, we use the data provided by the DRIAMS Dataset. The DRIAMS-dataset comprises a database of antimicrobials' resistance information and MALDI-TOF mass spectra. The data was collected in 2021 and the database was published in 2022. It contains mass spectra profiles from the most relevant isolates in clinical settings (bacteria) coupled with antimicrobial susceptibility phenotypes.[37]

In total, they collected $300,000$ mass spectra with more than $750,000$ antimicrobial resistance phenotypes from the daily clinical routines of four microbiological laboratories in Switzerland: University Hospital Basel (DRIAMS-A), Canton Hospital Basel-Land (DRIAMS-B), Canton Hospital Aarau (DRIAMS-C), the laboratory service provider Viollier (DRIAMS-D), which are all ISO/IEC 17025 accredited diagnostic laboratories. The University Hospital Basel and Conton Hospital Aarau both use the Microfex Biotyper LT/SH System MS, the laboratory service provider Viollier uses Microfex smart LS System, the Canton Hospital Basel-Land uses both Microfex Biotyper systems in parallel. Both systems are Microfex Biotyper Systems by Bruker Daltonics, which are widely used MALDI-TOF mass spectrometry systems in microbiological routine diagnostics in both North America and Europe. They were run in AutoXecute acquisition mode and serviced according to the manufacturer's standard. Although they use different laser gases, the systems use the same reference spectra database, which is why data from both systems was included. They use the Microfex Biotyper Database (MBT 7854 MSP Library, BDAL V8.0.0.0_7311-7854, research-use only) to identify the species of each mass spectrum.

The corresponding AMR profiles were collected in the same institutions in the same time frame. They used microdilution assays (VITEK 2, BioMérieux), minimum inhibitory concentration stripe tests or disc diffusion tests for determining the bacteria's resistance categories. With the exception of yeast, which was determined using Sensititre Yeast One (Thermofisher).

Furthermore, they use breakpoint measurements to interpret an organism as susceptible, intermediate, or resistant. Clinical breakpoints are pre-determined ranges used to classify the level of resistance[27], in this case they classify according to the EUCAST version using the most current breakpoint table update 56 (European Committee on Antimicrobial Susceptibility Testing) and CLSI (2015 M45; 2017 M60) recommendations.

To ensure equal quality of data at the different sites, empty spectra and calibration spectra are disregarded in further analysis.[38] Data analysis with the goal of AMR prediction necessitates that the dataset contains both, the mass spectra of bacteria, a biomolecule, and the bacterium's corresponding resistance labels. These are provided in the form of antimicrobial resistance profiles. The spectra and AMR profiles are stored in different databases in all 4 sites, therefore a matching procedure is undergone: The DRIAMS dataset is made up of files that contain the spectra without the corresponding genus and files that contain the species' genus and its AMR profiles, which they call the "laboratory report". For every MALDI-TOF mass spectrometry of a patient's probe, the Bruker Microflex system detects the species, which they then add to the 'laboratory report' as well as the AMR profiles, which are obtained in individual experiments. There is no link required between the spectrum file and the laboratory entry after the species is entered. Each entry in the laboratory report includes a code, which they refer to as 'sample-ID', it is the code that links an entry and a patient or their sample.

dataset	total
Escherichia coli - Ceftriaxone (E-CEF)	4961
Klebsiella pneumoniae - Ceftriaxone (K-CEF)	2860

Table 2.1: Number of samples by species-antibiotic combination in the DRIAMS dataset

This code may be ambiguous because there can be multiple experiments using one sample, or multiple samples of one patient. Furthermore, the spectra recorded by the Bruker Microflex systems were labeled with an ambiguous, that is, non-unique, code corresponding to the non-unique sample ID in the laboratory report. Therefore the utilised matching strategy to link mass spectra to their AMR profiles, is to create a unique identifier, in this case the tuple of the genus of the species and the sample ID. Using the genus and not the species in the identifier allows additional flexibility, as the Microflex Biotyper may provide a more accurate label, due to more extensive microbiological testing, which would mean that the species in the laboratory report differs from the species in the Microflex Biotyper System. They reanalyse the mass spectra with the University Hospital Basel Bruker library and determine the species and genus label to then use this information to create the unique identifier $(genus, code)$, a tupel, to pair the spectrum with the corresponding AMR profile. Multiple IDs only exist if the probe contains multiple genera, leading to multiple measurements. Cases in which the identifier was not unique were omitted. For details on each step of the matching procedure, please see the referenced paper [38].

In the scope of this thesis we limit ourselves to the species-antibiotic combinations: E. coli-Ceftriaxone and K. pneumoniae-Ceftriaxone. Both species are globally present and highly relevant, Ceftriaxone is a broad-spectrum beta-lactam antibiotic, therefore the combinations act as markers for broad-spectrum beta-lactam antibiotic resistance [38]. The whole DRIAMS dataset comprises 4961 samples for the species-antibiotic combination E. coli-Ceftriaxone and 2860 samples for K. pneumoniae-Ceftriaxone 2.1.

2.4 Preprocessing and Binning of Spectra

The spectra within the DRIAMS database have already undergone preprocessing steps. They were obtained in the Bruker Flex data format from the Bruker Flex machine. They were then preprocessed using the R package MaldiQuant58 v1.19 in the following way:

1. Variance stabilisation of intensities using the square-root method
2. Smoothing via Savitzky-Golay algorithm with a half-window-size of 10
3. Baseline correction by running 20 iterations of the SNIP algorithm
4. Calibration of the intensities using the total ion current (TIC), meaning the sum of all ionic currents of ions contributing to a mass-spectrum
5. Spectra trimming to mass-to-charge ratios within a range of 2,000-20,000 Da

Each spectrum consists of a set of n measurements which correspond to a mass-to-charge ratio r_n and the intensity of the ratio i_n. However, there is no fixed frame or range of m/z ratios which are measured for each species. Hence the dimensionality and spacing of the measurements differ. In the DRIAMS dataset, they include mass spectra in their raw version without any preprocessing as well as binned versions of the spectra. This is necessary because the used machine learning methods require feature vectors of fixed dimensionality. To remedy this, Weis et al.[38] bin the intensity measurements using a bin size of 3 Da in the range of 2,000 Da to 20,000 Da.The size 3 Da is big enough not to impede computational tractability but small enough to allow for separation of mass peaks. Binning is achieved by splitting the values within the range limits into disjoint, equal-sized bins along the m/z axis and then calculating the sum of the intensities of all measurements assigned to the same bin. This results in 6,000 bins in our case, we therefore work with feature vectors comprising 6,000 features.

Lastly, the categories of the AMR profiles, namely resistant (R), intermediate (I) and susceptible (S), are binarised during data input to create a binary classification scenario. The classes resistant and intermediate become class 1 and susceptible becomes class 0, as an antibiotic can be applied if the bacterium is susceptible but not in the intermediate nor the resistant case. Additionally, the intermediate category is usually classified as resistant in clinical practice because EUCAST v6–v8 has higher minimum inhibitory concentrations (MICs) meaning that a patient would have to be treated using a higher antibiotic drug concentration. Due to safety reasons, this is avoided to ensure an adequate safety padding when dealing with high antibiotic drug concentrations.[38]

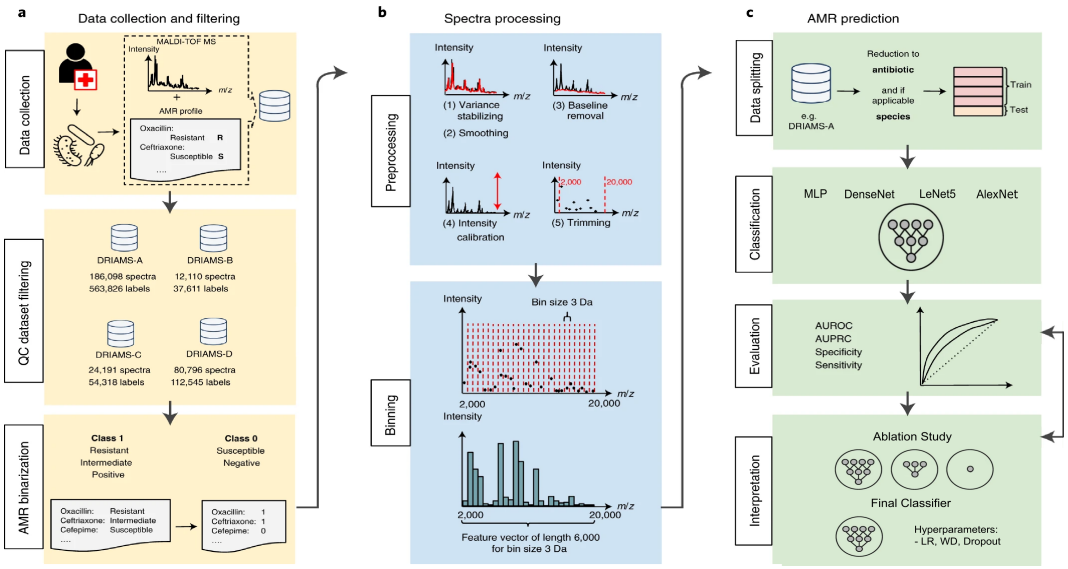


Figure 2.2: Modified from Weis et al. [38]. The original image was adapted to include our machine learning models and ablation study. This image contextualises our implementation into the clinical work-flow, beginning with the data collection and ending with the result interpretation.

3 Related Work

Battling the AMR crisis is critical to uphold the state of the art of modern medicine[3], in this context artificial intelligence has emerged as a powerful tool. One fostering factor is the availability of big amounts of data to feed into AI algorithms.[1] However the research landscape concentrating on practical applications of resistance predictions such as diagnosis and treatment remains sparse and mainly unsuccessful.[1] One significant step in the direction of practical application is the use of fast data retrieval methods, such as MALDI-TOF, therefore we focus on machine learning predictions for AMR using bacterial MALDI-TOF spectra.

3.1 Traditional Machine Learning Methods: RF, SVM, DT, KNN

The body of research utilises a group of common machine learning methods for resistance prediction of specific bacterium-drug combinations to make comparison possible. As Weis et al.[41] show, KNN, SVM, LR, DT and RF make up for more than 70% of experiments. In this way, Wang et al. [35] implement DT, KNN, RF, and SVM to predict methicillin-resistant S. aureus resulting in a leading AUROC value of 0.8450 (RF). Accordingly, Feucherolle et al. [9] successfully perform predictions using LR, RF and naive bayes for bacteria C. coli and C. jejuni combined with Ciprofloxacin, or Tetracycline reaching AUROC values of 0.87, 0.83, and 0.88, 0.80 respectively.

Furthermore, many approaches try increasing the performance of said algorithms by applying changes to the input spectra, such as mass selection, persistence transformation[39], binning of spectra[44] or train-test split stratification by using an amount of k clusters and labelling each cluster according to phylogeny. Weis et al. [40] use the latter together with classifiers like LR and GP-PIKE, the Gaussian process (GP) in combination with the Peak Information Kernel (PIKE). PIKE assesses the similarity between MALDI-TOF spectra of unfixed lengths by creating a feature map from the space of functions into $L^2(\mathbf{R})$. In this case, the function is a sum of Dirac delta functions, one function per m/z value, with scale factors that take into account the intensity (height) of the peak (Dirac delta functions are approximated by $L^2(\mathbf{R})$ functions). Lastly, they calculate the inner product of $L^2(\mathbf{R})$ space of two spectra, which were transformed into $L^2(\mathbf{R})$ by the feature map. This results in the similarity measure. [39] This method also enables the calculation of uncertainty measures, which is an important variable for clinical application [40]. A general shortcoming of the mentioned approaches is that the acquired models are not applicable to other prediction scenarios, meaning changed variables like antibiotic-species combinations, locations of data collection, Mass Spectrometer manufacturer or even train-test splits. More precisely, when using the GP-PIKE, the optimal parameters of one data split applied to another data split lead to convergence issues and consequently varying performance scores; added methods like hierarchical stratification fail to reduce the standard deviation between the results of

different data splits. One related problem is that data sets can be unbalanced and contain spectra of isolates that are not sufficiently representative. The presented methods are not able to compensate for these problems and, therefore, do not yet provide clinically applicable solutions for rapid resistance predictions.

paper	bacterium	antibiotic	classifier	metric	performance
Wang et al.[35]	S. aureus	Methicillin	DT, KNN, RF, SVM	AUROC	0.74, 0.82, 0.84,0.81
Feucherolles et al.[9]	C. coli	Ciprofloxacin,	RF, LR, Naive Bayes		0.87
		Tetracycline	RF, LR, Naive Bayes		0.83
	C. jejuni	Ciprofloxacin	RF, LR, Naive Bayes		0.88
		Tetracycline	RF, LR, Naive Bayes		0.8
Weis et al.[40]	E. coli	Ciprofloxacin	GP-PIKE, LR	AUPRC	0.67±0.03, 0.72±0.03
		Ceftriaxone	GP-PIKE, LR		0.64±0.05 , 0.72±0.04
	K. pneumoniae	Ciprofloxacin	GP-PIKE, LR		0.48±0.15, 0.55±0.11
		Ceftriaxone	GP-PIKE, LR		0.64±0.08, 0.77±0.07
	S. aureus	Ciprofloxacin	GP-PIKE, LR		0.34±0.03, 0.40±0.10

Table 3.1: Overview of the mentioned publications of traditional machine learning methods and corresponding classifier performances.

3.2 Neural Networks

Another essential and frequently used tool in medical research is artificial neural networks[45]: Artificial neural networks (ANN) simulate the sensory processing of the human brain. ANNs consist of a number of interconnected model neurons, also known as perceptrons. Each neuron weighs and then adds up the inputs it receives from other units or external sources. Similar to real neurons, if the action potential, in ANNs, the calculated sum, rises above a certain threshold, the neuron is activated, and consequently, potential, in our case data, is transmitted. Therefore, the classifier's model neurons output becomes 1 instead of 0 when the threshold is met; a simplified corresponding function can be found in Figure 3.1.[16]

$$f(x;\theta) = H(\beta^T x + b) \tag{3.1}$$

The equation 3.1, which is reproduced from Weis [42] defines the perceptron's behaviour. f calculates the neuron's output using input feature vector x and the model parameters $\theta = (\beta, b)$, namely weights and bias. The sum over all inputs x_i, weighted by their corresponding weights β_i is calculated. The bias b is added, and the chosen linear threshold activation function is applied.As $H \in \{0, 1\}$, H discretises the neuron's output, values subceeding the threshold become 0, and those reaching the threshold become 1. [16, 42]

The ANN is trained and, therefore, learns to solve tasks; this process involves adjusting parameters such as weights and biases to minimise the dissimilarity between the predicted output and the actual true output; this is realised by backpropagation; After the input has been passed through the network in a feed-forward manner, the discrepancy of the calculated result at the output layer and the true result is assessed using a loss function. Moving backwards through the network, the derivatives of the loss function are calculated with respect to each weight, which is done in only one pass using the chain rule. The derivative is then used to update the weights in a way that minimises the error.[43]

Basic neural networks consist of an input layer, a hidden layer and an output layer; in the case that more hidden layers are added, the network is considered a deep artificial neural network (DNN).[6]

Convolutional neural networks are one of the most popularly applied versions of ANNs. Accordingly, CNNs are made up of neurons with learnable weights and biases. However, each neuron is merely connected to a subset of neurons from the previous and following layers. Due to the sparsity and weight sharing created during convolution, training is made efficient. The main layers within such a network are the convolutional layer, the pooling layer and the fully connected layer. [2] Convolution is used for feature extraction by sliding a filter across different areas of the data. For each position, a dot product calculates one value of the output feature map; the filter is moved by one step-size each, this is defined by the stride.[19]

Weis et al.[38], as well as others, test deep learning approaches compared to more restricted methods: They compare LR, LightGBM, MLP and find that LightGBM and MLP are the best-performing classifiers in terms of AUROC (AUROC>0.7) for their species-drug combinations. They also present a multimodal learning approach over proteomic and chemical features, for

paper	bacterium	antibiotic	classifier	metric	performance
Weis et al.[33]	E. coli	Cefepime	ResMLP/ LightGBM	AUROC	0.87/ 0.71
	S. aureus	Oxacillin	ResMLP/ LightGBM		0.94/ 0.78
Fu et al.[11]	P. aeruginosa	Tobramycin/ Cefepime/ Meropenem	DNN	AUROC	0.77/ 0.85/ 0.9
Wang et al.[36]	Enterococcus faecium	Vancomycin	CNN	AUROC	0.887

Table 3.2: Overview of mentioned publications of neural network methods and corresponding classifier performances.

which they implement a siamese neural network and an MLP with residual skip-connections.[33] They observe that the multimodal ResMLP methods outperform the conventional MLP as well as the siamese neural network, as it demonstrates greater consistency across drugs.

Additionally, Fu et al.[11] design a deep neural network synergised with a strategic sampling approach which outperforms more conventional classifiers in AMR prediction of P. aeruginosa (AUROC: 0.77-0.9). Similarly, the CNN classifier in the publication "Efficiently predicting vancomycin resistance of Enterococcus faecium from MALDI-TOF MS spectra using a deep learning-based approach"[36] successfully predicts vancomycin-resistant Enterococcus faecium (AUROC: 0.887). It is important to note that AUROC values are not comparable across different datasets with variable data distributions.[4]

4 Methods

4.1 Data Preparation

In previous work by Caroline Weis et al.[38, 40], the focus is set on more basic and traditional machine learning algorithms. Therefore, the classifiers, as well as the data retrieval, were implemented using Scikit-learn.

However, we want to improve generalisation by implementing more sophisticated ML approaches; this is why we choose to use PyTorch libraries for the implementational add-ons we provide. PyTorch is a tensor library which is optimised for high-performance tasks such as deep learning on GPU and CPU [21]. PyTorch is suitable for our task, as we work with large amounts of data, necessitating the implementation to be fast and efficient. We also want to analyse and customise the classifiers, which can be done smoothly using PyTorch. Consequently, we retrieve the data utilising the base implementation using Scikit-learn libraries and then transform it into a format useful for PyTorch in the following way.

4.1.1 Splitting

We ensure that samples of a specific patient case cannot be separated into train and test split. They need to remain on one side of the split as sample measurements of the same infection may be very similar and, therefore, could lead to information leakage from training to testing. Furthermore, we randomly split the data along a train-test ratio of 80, meaning that the training dataset comprises 80%, and the test dataset is 20% of the whole dataset. We make the randomness in splitting reproducible by introducing ten different random seeds. In order to decrease the data imbalance or to prohibit unrepresentative splits, we apply a stratification along class and species, which guarantees similar prevalence values. We follow the same procedure, regardless of whether train- and test-domains differ.[38]

4.1.2 PyTorch Tensors

To efficiently store and load the spectral data, this includes spectra and their corresponding labels, we use PyTorch's tensor data format. In mathematics, a tensor generalises the concept of scalars, vectors and matrices to higher dimensions.[21, 29] Analogously, tensors used in data science are multi-dimensional matrices or arrays containing elements of the same data type.

We utilise PyTorch tensors to store our spectral data by creating a tensor of tensors, each tensor representing one spectrum. In the work we extend, the pre-existing retrieved data is represented in the form of Numpy arrays of type float and length 6,000, as we use the bin size 6,000. We create the PyTorch tensors by transforming the corresponding Numpy arrays using the method torch.tensor(), this copies the data from the array and initialises the tensor with the copied data.

Our further calculations necessitate tensors of the type float; we, therefore, use *Tensor.to()*, which converts a tensor's data type to another data type, in this case, 32-bit floats. Accordingly, our labels are stored within a tensor of 64-bit unsigned integers.

```
X_train = torch.tensor(X_train).to(torch.float32)
X_test = torch.tensor(X_test).float()
y_train = torch.tensor(y_train)
y_train = y_train.to(torch.int64)
```

Furthermore, when a GPU is available, we can efficiently transfer our data from the CPU to the given GPU and vice versa also using *Tensor.to()*.

```
#transferral of tensors to GPU if GPU is available
if torch.cuda.is_available():
      data.to("cuda")
#transferral of tensors to CPU
data.to("cpu")
```

4.1.3 Spectral Dataset and Dataloaders

The feature and label tensors are then combined into a *SpectralDataset*, which is a custom implementation of the *torch.utils.data.Dataset* class and stores the spectrum-label pairs[28]. This allows us to index data pairs and get the number of 0 and 1 labels within the dataset.

```
First spectrum-label pairs in train- and test-dataset
(tensor([ 0.2686, -0.0031, -0.8370,  ..., -1.1238, -1.1623, -1.1571]), tensor(0))
(tensor([-0.8388, -0.7172, -0.8135,  ..., -0.4992, -0.2588, -0.8266]), tensor(0))
```

Figure 4.1: This terminal output shows how two exemplary spectrum-label pairs are stored in our implementation.

The *torch.utils.Dataset* class is commonly used together with the *torch.utils.data.Dataloader* class, which wraps an iterable around the dataset [28]. This allows for parallelisation of the data loading process due to automatic batching, resulting in a reduction of needed memory and higher speed[25]. We create dataloaders for the training set, the validation set and the testing set, with a default batch size of 32.

A batch size of 32 is the default value in the Keras deep learning API and provides computational efficiency as well as good performance across many datasets, therefore we deem this a good fit as a starting point[8].

```
trainDataset = SpectralDataset(X_train, y_train)
trainloader = torch.utils.data.DataLoader(trainDataset, batch_size=bs, shuffle=False)
testDataset  = SpectralDataset(X_test, y_test)
testloader = torch.utils.data.DataLoader(testDataset, batch_size=bs, shuffle=False)


print("labels DRIAMS-A train-dataset, seed: 344")
trainDataset.label_dist(trainloader)
print("labels DRIAMS-A test-dataset, seed: 344")
testDataset.label_dist(testloader)
```

```
labels train-dataset:DRIAMS-A , seed: 344
# 0-labels: 3096
# 1-labels: 358
labels test-dataset: DRIAMS-A, seed: 344
# 0-labels: 765
# 1-labels: 83
```

Figure 4.2: This code snippet shows the creation of datasets and dataloaders for the training and testing scenario, further it shows how to calculate the number of resistant and susceptible samples within a dataset, as well as the corresponding output.

4.2 Models

The goal of this work is to contribute toward increasing the performance results of the DRIAMS dataset, especially in terms of generalisation performance. Limiting factors are the imbalance in the dataset and the relatively small number of samples per species-antibiotic combination[30]. To overcome these challenges, we set the final goal of exploring the potential of transfer learning on the DRIAMS dataset. We do this by establishing the performance of relevant CNN models to create a suitable pre-trained model whose information shall be transferred into a new task. We choose to concentrate on CNNs because of their great ability in image classification. In addition, CNNs are commonly used in transfer learning models[30] and to our knowledge, have not yet been explored in experiments within the scope of the DRIAMS dataset.

4.2.1 MLP Classifier

To establish the baseline performance for our models, we recreate the MLP classifier mentioned in "Direct antimicrobial resistance prediction from clinical MALDI-TOF mass spectra using machine learning" [38]. Multi-layer perceptrons (MLP) are simple neural network classifiers, consisting of multiple stacked perceptrons. In the following we will refer to their implementation as *maldi_amr*. We implement our models using PyTorch libraries, wheras the *maldi_amr* implementation uses Scikit-learn libraries. This necessitates that we are able to achieve similar performance when using the same model architecture in PyTorch. Therefore We initialize the PyTorch MLP with the default parameters of Scikit-learn MLPs:

```
def __init__(self, input_size=6000, hidden_layer_sizes=(256,64), output_size=1, activation='relu', alpha=
                                                0.0001, learning_rate_init=0.001, max_iter=500,
                                                shuffle=True, random_state=None, scaler = None):
```

Furthermore we implement gridsearch functionality, to replace the Scikit-learn GridsearchCV object used in *maldi_amr*. We use the parameter grid as shown in Table 4.1:

optimized metric	epochs	learning rate	weight decay	hidden layers
validation AUROC	100	0.001	0.0001	(256,64),(256,128)
				(512,128,64),(512, 256, 128)

Table 4.1: The range of parameters used for parameter search and optimisation of the PyTorch MLP classifier.

We implement early stopping with a patience value of 10 and a tolerance of $1e$-4, hence if the training loss decreases by less than $1e$-4 for 10 epochs in a row, then training is stopped early on, meaning before all epochs were iterated over. In order to explore the performance of CNN networks on the DRIAMS dataset, we select the most prominently used convolutional neural networks (CNN). The networks are ordered by date of publication ascendingly.

4.2.2 LeNet5

The LeNet5 deep learning architecture was designed by Yann LeCun et al. in 1998 and can be considered a standard template for other convolutional networks following this publication. Its popularity stems from the simple and straightforward architecture as well as its usefulness in image classification due to the multilayer convolutional neural network architecture.[10] Namely, LeNet5 comprises two convolutional layers, followed by three fully connected layers. Each convolutional layer contains a rectified linear unit (ReLU) and pooling mechanisms.

4.2.3 AlexNet

The AlexNet (2012) architecture comprises five convolutional layers and three fully-connected layers. The first two and the last convolutional layers are followed by a max-pooling layer; dropout is used in the fully connected layers as a means to reduce overfitting. Nonlinearities are introduced in the form of rectified linear units to increase training speed. Pooling layers consist of multiple pooling units assembled with a distance of s pixels; each pooling unit covers a space of $z * z$. For CNNs $s = z$ is commonly used, but in the AlexNet architecture it holds that $s < z (s = 2, z = 3)$, resulting in overlapping pooling layers.[15]

4.2.4 VggNet

VggNet (2014) is a very deep neural network which commonly facilitates good performance and generalisation using simple pipelines through depth. Depending on the architecture, either 13 or 16 convolutional layers are followed by three fully connected layers. Non-linearities are implemented into every hidden layer, batch normalisation is incorporated into every convolutional layer, and dropout is added to the first two fully connected layers.[31]

4.2.5 DenseNet

The core idea of the DenseNet (2018) architecture is shorter connections between layers close to the input- and those close to the output layer, to make training deeper, more accurate, and more efficient. DenseNet is a dense convolutional network, meaning that each layer is directly connected to every other layer with matching feature map sizes in a feed-forward fashion. A pivotal difference to standard neural networks is that in a network of L convolutional blocks, each layer l receives the feature maps of all preceding convolutional blocks $0 - (l - 1)$, layer l adds its' own feature maps and passes them to the subsequent layers $(l + 1) - L$ resulting in $(L * (L + 1))/2$ feature maps.[12]

Model	Layers		Unique Feature	Year
	Conv	FC		
LeNet5	2	3	pioneer model	1998
AlexNet	5	3	overlapping pooling layers	2012
DenseNet	99	1	block structure, direct connections	2018
VGGNet16	13	3	very deep neural network	2014
VGGNet19	16	3	very deep neural network	2014

Table 4.2: This is an overview of the model architectures we implemented and use in the scope of this thesis. It displays the name, the number of convolutional layers (Conv), the number of fully connected layers (FC), a unique feature of the model architecture and the year the specific architecture was published.

4.3 Evaluation

We assess model performance based on the following metrics: Area Under the Receiver Operating Characteristic Curve (AUROC), Area Under the Precision-Recall Curve (AUPRC), and accuracy. For a deeper understanding, we also introduce the metrics true positive rate, false positive rate, precision and recall. Receiver operating characteristic (ROC) graphs plot the classifier's true

$$TPR = \frac{TP}{P} \quad \text{Recall} = \frac{TP}{P}$$

$$FPR = \frac{FP}{N} \quad \text{Precision} = \frac{TP}{TP + FP}$$

Figure 4.3: Performance metrics for binary classification, with TP denoting the number of true positives, FP denoting the number of false positives, P denoting the actual number of total positives, N denoting the actual number of total negatives in the binary classification task. Therefore TPR defines the true positive rate which is the portion of positives which were correctly predicted as positive. FPR defines the false positive rate, which is the portion of negatives that were falsely predicted as positive.

positive rate (y-axis) against the classifier's false positive rate (x-axis). Classifiers assign a discrete value between 0 and 1, by introducing a decision threshold, by default 0.5, which classifies the indiscrete prediction values between 0 and 1. The ROC curve is built by shifting the threshold value from $-\infty$ to ∞ and plotting the corresponding results. The used metric, the area under the receiver operating characteristic (AUROC), is the integral of the ROC curve and can, therefore, be interpreted as the probability of successfully classifying a pair of samples from both classes.[38] [42] Similarly, precision-recall curves plot the classifier's precision scores (y–axis) against the clas-

$$\text{AUROC} = \sum_{i=1}^{n-1} \frac{(FPR_{i+1} - FPR_i) \times (TPR_i + TPR_{i+1})}{2}$$

Figure 4.4: FPR_i and TPR_i denote the false positive and true positive rate at the i-th threshold.

sifier's recall scores (x–axis) for varying threshold values between $-\infty$ and ∞. The area under the precision-recall curve can be understood as the classifier's ability to correctly classify samples from the minority class whilst also minimizing false discoveries.[38] [42] The accuracy score is

$$\text{AUPRC} = \sum_{i=1}^{n-1} (R_{i+1} - R_i) \times \frac{P_i + P_{i+1}}{2}$$

Figure 4.5: R_i and P_i denote the Recall and Precision at the i-th threshold.

the division of the number of correctly classified samples by the total number of samples. To

$$\text{Accuracy} = \frac{TP + TN}{P + N}$$

Figure 4.6: As mentioned above, TP and TN are the number of true positives and negatives respectively. P and N are the total number of actual positives and negatives.

contextualise these values, an AUROC score of 1.0 is perfect, whereas a score of 0.5 equals the classification ability of a random classifier. An AUPRC score of 1.0 is perfect, whereas a random classifier will achieve the class ratio of the minority class.

4.4 Weights and Biases

We employ Weights & Biases for experiment logging and result visualisation. This tool enables us to track training loss, validation loss, AUPRC, AUROC, f1- and recall- score of every epoch, allowing an in-depth analysis of model performance. We predominantly use W&B sweeps to explore the potential of model performance through automatic hyperparameter search within a particular grid 4.3.

We perform five runs per sweep, meaning we run every scenario five times. After establishing a detailed overview of the models' generalisation performance, we identify the enormous amount of parameters as a possible drawback. Therefore, we conduct an ablation study to further inspect the models' behaviour.

optimized metric	batch size	epochs	learning rate	weight decay
validation AUROC	32	5, 10, 15, ... ,45, 50	$0.01 - 0.000001$	$0.01 - 0.000001$

Table 4.3: The range of parameters used in W&B parameter search and optimisation.

4.5 Ablation study

Ablation studies are used to analyse the performance of a machine learning system and the impact of its components by removing specific components from the system. This idea is derived from neuroscientific ablation studies, in which parts of the brain are damaged or removed in order to understand the brain regions' influence on the brain's ability to perform certain tasks.[20]

Time constraints limit our ablation study to LeNet, AlexNet model architectures. We ablate AlexNet and LeNet by eliminating one layer, either convolutional or fully connected, at a time.

5 Experiments

5.1 Experimental Setup

We perform the baseline experiments on a system with two Intel(R) Xeon(R) CPU E5-2687W v3 @ 3.10GHz processors, each with ten cores and two threads per core, resulting in a total of 40 logical CPUs. However, the amount of data (144.84 GB 2.3) and the depth of the initial models drastically restrict the speed of model training on this suboptimal setup. Due to our time constraints, we decided to switch to an NVIDIA Tesla V100 GPU, which is available through Google Colab's allocation of resources. Tesla V100 features 640 Tensor Cores and 5,120 CUDA cores, and the system provides 51GB System RAM and 16GB GPU RAM. This significantly upgrades our system's computational power and efficiency, especially for deep learning tasks.

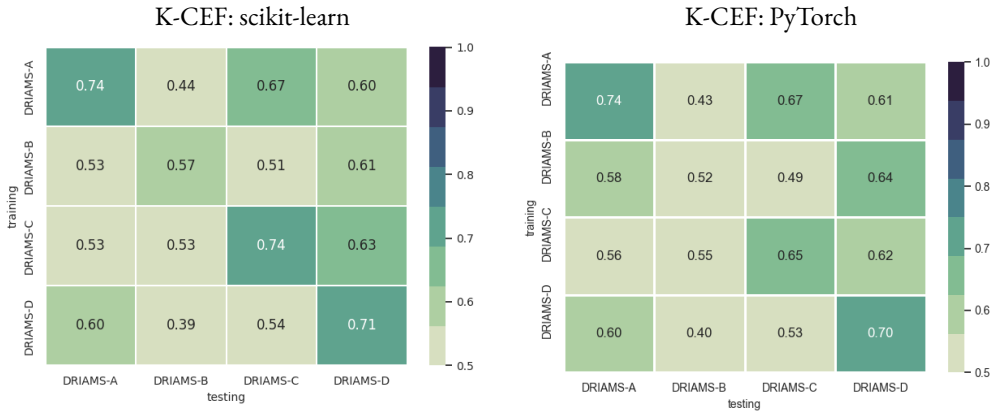
As this thesis has the objective of extending the work displayed in "Direct antimicrobial resistance prediction from clinical MALDI-TOF mass spectra using machine learning" [38], with a focus on **generalisation**, we perform our experiments according to their site-specific training and testing on sites DRIAMS-A through DRIAMS-D: We pair every train dataset D_{train} in DRIAMS-A—D with every test dataset D_{test} in DRIAMS-A—D; therefore, we create 16 Tuples (D_{train}, D_{test}), our 16 train-test scenarios S. For every scenario $s_i \in S$ we train, test and evaluate a specific model using the same ten different shuffled stratified train–test splits T.[38] For every split $t_i \in T$ in every scenario s_i, we run a grid search with 5-fold cross-validation to retrieve the most valuable hyperparameters in terms of AUROC scores. In some cases, the training set cannot be divided into 5-folds; we then switch to 3-fold cross-validation.

Before training, we prepare our data akin to the description in section 4.1. We apply the widely known evaluation metrics AUROC, AUPRC and Accuracy to each experiment.

5.2 Baseline Experiment

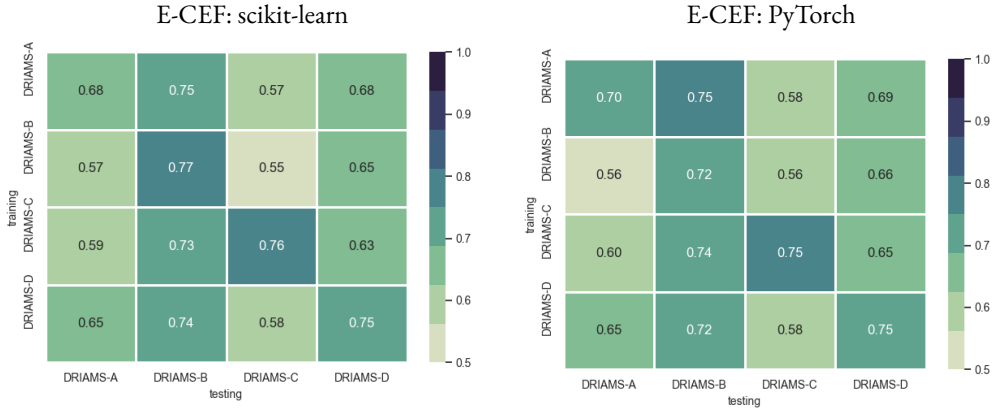
We evaluate the Pytorch MLP classifier for all mentioned train-test scenarios. This experiment sets the baseline performance, which we use to interpret and analyse the results from experiments on other classifiers.

The comparison of the two MLP classifiers' results for the species *Klebsiella Pneumoniae*, depicted in 5.1, shows that we achieve very similar scores, the AUROC performance deviates by a value of ± 0.02 in most cases. However, the AUROC performance exhibits larger performance deviations in two scenarios: the scenarios DRIAMS-B—DRIAMS-B and DRIAMS-C—DRIAMS-C deviate by -0.05, -0.09, respectively. We attribute the causes for this deviation to minor differences between the implementations of the two frameworks (e.g.: numerical precision, optimisation) as well as differences in our implementation of the grid search functionality compared to that of scikit-learn.



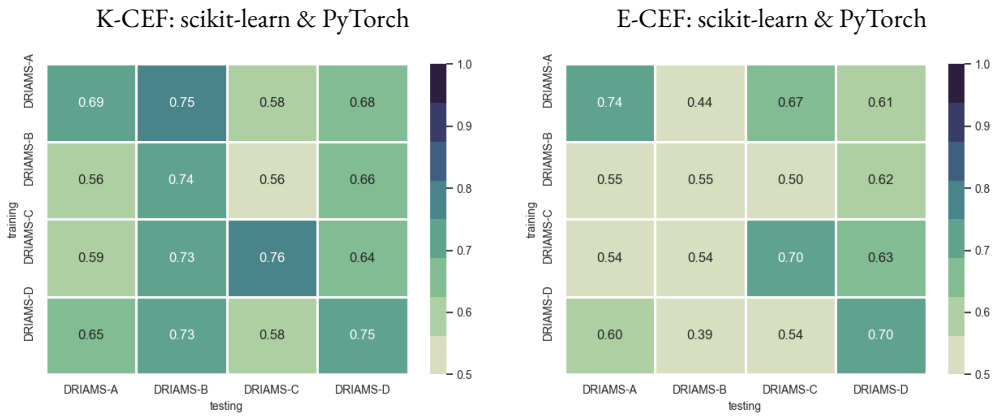
(a) (b)

Figure 5.1: Mean AUROC performance of the same 10 random train-test splits for Klebsiella Pneumoniae (K) and Ceftriaxone (CEF) implemented using scikit-learn (left) and PyTorch (right) libraries. The mean performance for all train-test scenarios for the scikit-learn and PyTorch classifier reaches 0.584 and 0.581, respectively.



(a) (b)

Figure 5.2: Mean AUROC performance of the same 10 random train-test splits for Escherichia Coli (E) and Ceftriaxone (CEF) implemented using scikit-learn (left) and PyTorch (right) libraries. The mean performance for all train-test scenarios for the scikit-learn and PyTorch classifier reaches 0.666 and 0.666, respectively.



(a) (b)

Figure 5.3: Mean AUROC performance of the same 10 random train-test splits for Klebsiella Pneumoniae (K) (left) and Escherichia coli (E) (right) in combination with Ceftriaxone (CEF) using results from both implementations, scikit-learn and PyTorch libraries. The mean performance for all train-test scenarios for K-CEF and E-CEF reaches 0.5825 and 0.666, respectively. This performance matrix exhibits the mean performance of the implementation-specific performance matrices.

The same is true for the results in 5.2, which show the baseline experiment for the species *Escherichia Coli*. The only noticeable deviation occurs in the DRIAMS-B—DRIAMS-B scenario, here the PyTorch implementation reaches an AUROC score of 0.72 instead of 0.77 (scikit-learn), therefore deviating by 0.05.

Both classifiers perform significantly worse on data of the bacterium *Klebsiella pneumoniae* compared to data of *Escherichia coli*. More precisely, both classifiers only reach a mean AUROC performance over all train-test scenarios of 0.58 for K-CEF, which is 0.08 less than for E-CEF for which both classifiers achieve a mean AUROC of 0.66. A reason for this could be that the total amount of samples for K-CEF (2,860 samples) is significantly smaller than the total amount of samples for E-CEF (4,961 samples) 2.1.

The MLP model architecture is able to generalise the best when the model is trained on the DRIAMS-A dataset for both implementations and both species. Furthermore, both implementations achieve the best predictive scores for K-CEF for the train-test scenario DRIAMS-A–A, achieving 0.74 in terms of AUROC. This suggests that the model either merely learns the noise in the training dataset or that the site-specific datasets differ too extremely, as the classifier is incapable of yielding similar performance values for scenarios in which the training and testing sites are different.

However, this is not the case for predictions on the combination E-CEF. Here, the best predictive performance is achieved in the scenario DRIAMS-B–B using the scikit-learn implementation, achieving an AUROC of 0.77. For the PyTorch implementation per contra, the best AUROC scores can be observed in scenarios DRIAMS-A–B and DRIAMS-C–C, having a value of 0.75. Interestingly, a multitude of scenarios exhibit similar performances using the E-CEF data, especially scenarios tested on DRIAMS-B or DRIAMS-D, which consistently yield AUROC scores above 0.6. Again, we attribute this to the availability of more data.

5.3 Neural Networks on Datasets from Multiple Domains

5.3.1 Model exploration

We explore the potential performance increase, compared to the baseline, of the specific model architectures through hyperparameter optimisation. We use W&B to optimise the hyperparameters for the scenarios we find relevant. In this context, we term scenarios relevant, in which the MLP classifier is not able to generalise and predict more accurately than a random guess (AUROC = 0.5) for K-CEF. We also consider scenarios, for which the performance of the classifier is only slightly better than 0.5. Therefore our experiments include scenarios: DRIAMS-A–B, DRIAMS-B–[A,C,D], DRIAMS-C–[A, B, D], DRIAMS-D–[C,B]. W&B enables us to visualise this experiments in Figures 5.4, 5.5, 5.6, 5.7. These parallel coordinate plots show the results for every model architecture for this experiment for both species *Klebsiella pneumoniae* and *Escherichia coli*; we run one so-called sweep for every scenario, which comprises five runs. During one sweep, W&B performs the hyperparameter optimisation from run to run. One line in the plot represents one W&B run, the selected parameters and the model's performance in terms of AUROC and AUPRC.

We use the range of prediction scores to assess the performance of classifiers MLP, AlexNet, LeNet5, DenseNet and VGGNet. Accordingly, we find that none of the models achieves better

performance than the MLP classifier. We notice that most W&B runs using all classifiers accumulate at AUROC scores in the range of [0.3; 0.7], disregarding the few outliers.

The very deep VGGNet architecture and the DenseNet perform slightly worse than the other classifiers with a range of [0.32; 0.62] and [0.25; 0.7]. The MLP classifier achieves scores within [0.31; 0.77]. The LeNet5 classifier reaches an interval of [0.33; 0.80]. Similarly, the AlexNet reaches slightly better scores with a range of [0.38; 0.81].

In the parallel coordinate plot for the MLP classifier 5.4, it is noticeable that significant differences in the AUROC score occur depending on the specific train-test scenario. The runs within one scenario achieve very similar results. Therefore, the achieved AUROC score ranges for a sweep do not overlap as much as, for example, in the AlexNet 5.5 or LeNet5 plot 5.6.

The same effect, only less blatantly, can be observed in the parallel coordinate plot for the DenseNet classifier 5.8. This implies that the model is incapable of performing similarly for various scenarios, meaning it does not generalise well.

As the scales for the plots are adapted to the achieved scores, one can see that the AlexNet 5.5 and LeNet5 5.6 classifier perform within the same range. The colour of the lines within both plots is predominantly purple, showing that mostly values between 0.3 and 0.6 are achieved. However, the AlexNet performs above-average for scenarios with the test-site DRIAMS-B, whereas the LeNet5 classifier achieves comparably higher AUROC scores on the train-site DRIAMS-C.

Furthermore, the AUROC values of the VGGNet experiment accumulate independently from the train-test scenario, we observe predominantly orange lines in the parallel coordinate plot 5.7. However, the AUROC scores are very much below-average compared to the other classifiers, the scale begins at 0.15 and only reaches 0.65.

We further analyse and detail model behaviour beneath the corresponding parallel coordinate plots. The corresponding W&B report can be found here, additionally, we have attached the corresponding loss plots to the Appendix 6.

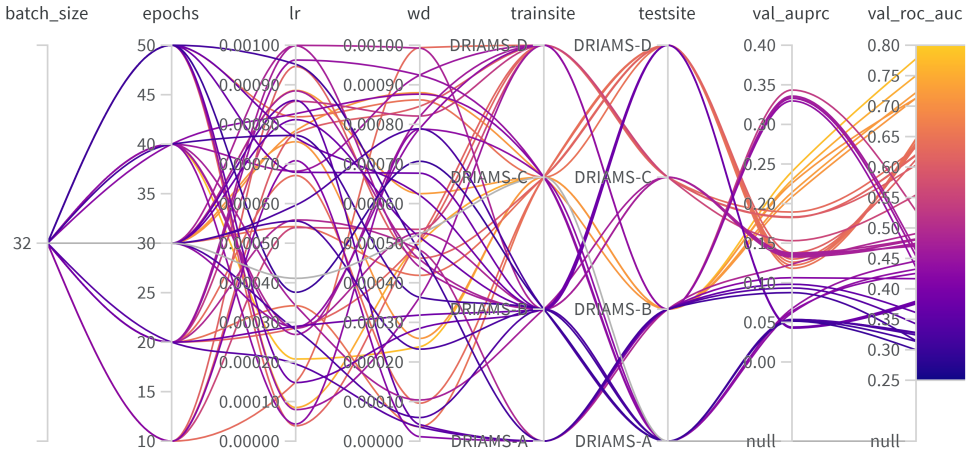


Figure 5.4: This parallel coordinate plot shows the results of the model exploration of the MLP classifier. The logged values of the runs for which parameter search and optimisation was carried out by W&B: Batch size, number of epochs, learning rate (lr), weight decay (wd), train dataset (trainsite), test dataset (testsite), AUPRC (val_auprc), AUROC (val_roc_auc). The MLP architecture performs the best in the scenario DRIAMS-C–B with AUROC values between 0.71 and 0.77. It also yields good performances for DRIAMS-C–D, showing that the model can generalise well when trained on the domain DRIAMS-C to two other sites, namely DRIAMS-B and DRIAMS-D. The opposite scenario, DRIAMS-D–C, achieves very similar results to DRIAMS-C–D; however, the model is not able to generalise from DRIAMS-B to DRIAMS-C, as it achieves AUROC scores slightly below 0.5. No clear trend is noticeable for the hyperparameters for well performing runs, learning rates range from 0.0002 to 0.0008 and the weight decay settles between 0.0002 and 0.0009, which is a big range. In general, the MLP model yields consistent performances for each scenario, meaning that the AUROC performances within one scenario deviate by less than 0.1 in most cases, albeit the hyperparameters being changed significantly. However, the performance gap between the scenarios DRIAMS-C–B and DRIAMS-C–A or DRIAMS-B–A adds up to 0.3. This significant dissimilarity can be attributed to overfitting, as shown in figure 5; the model learns the noise in the training data instead of underlying patterns, and therefore, high AUROC values occur due to similarities in the validation and training dataset.

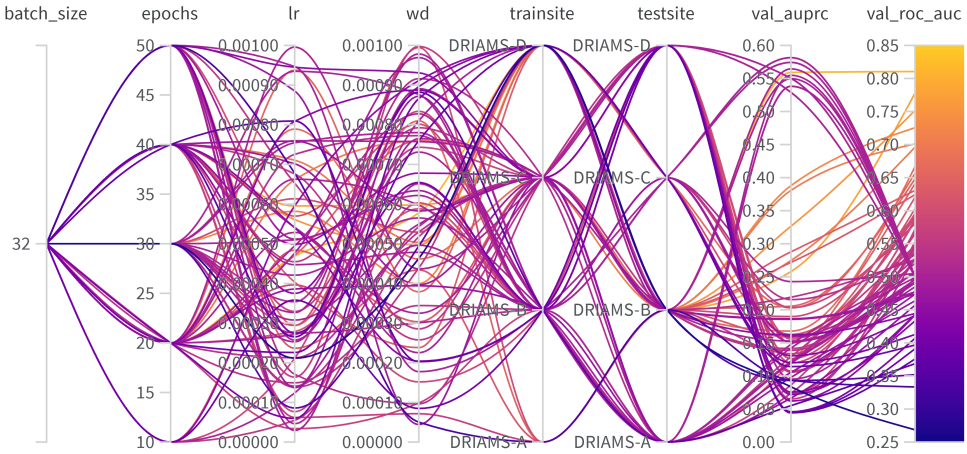


Figure 5.5: This parallel coordinate plot shows the results of the model exploration of the AlexNet classifier. The logged values of the runs for which parameter search and optimisation was carried out by W&B: Batch size, number of epochs, learning rate (lr), weight decay (wd), train dataset (trainsite), test dataset (testsite), AUPRC (val_auprc), AUROC (val_roc_auc). The best performance is achieved for the scenario DRIAMS-D–B and the species Escherichia coli with an AUROC score of 0.81. Interestingly, the worst performance is also achieved for the scenario DRIAMS-D–B but with the species Klebsiella pneumoniae with an AUROC score of 0.2685. This vast gap can be attributed to less data for the latter species, nevertheless performance differences of more than 0.2 in terms of AUROC can be observed in most sweeps, meaning in the cases where only the hyperparameters are varied, not the species. Consequently, the model is very sensitive to hyperparameters, which is common for deep neural networks, especially to weight decay and epochs, as shown in the W&B report. The classifier outcomes benefit from mid-range weight decay as well as learning rates, meaning between 0.0004 and 0.0007. Aside from the performance fluctuations, the model reaches good AUROC scores using DRIAMS-B as the validation dataset and any other dataset as the training dataset. The AlexNet cannot yield increases in generalisation performance in the other cases; one cause is that the validation loss remains very unstable, as shown in figure 7.

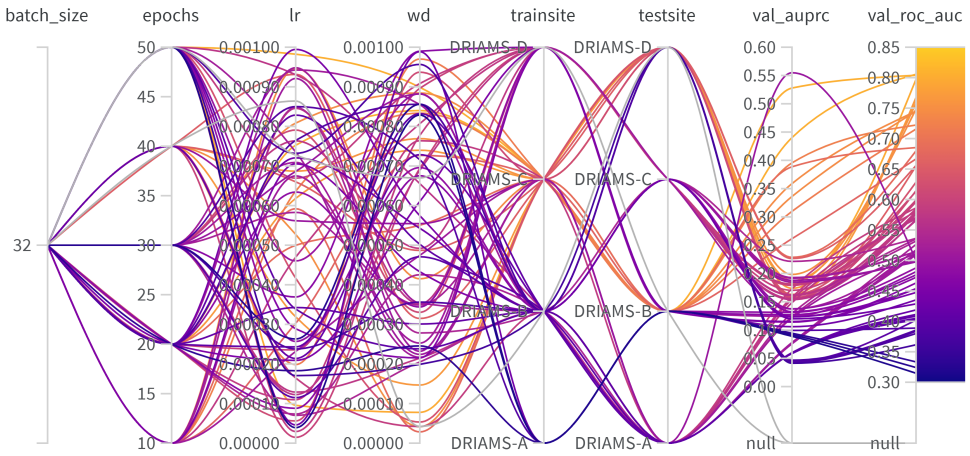


Figure 5.6: This parallel coordinate plot shows the results of the model exploration of the LeNet5 classifier. The logged values of the runs for which parameter search and optimisation was carried out by W&B: Batch size, number of epochs, learning rate (lr), weight decay (wd), train dataset (trainsite), test dataset (testsite), AUPRC (val_auprc), AUROC (val_roc_auc). The best performance in terms of AUROC is achieved for scenario DRIAMS-C–B with 0.8. Comparably high AUROC scores are also achieved in the scenario DRIAMS-C–D (0.68) and the scenario DRIAMS-C–A outperforms other scenarios tested on DRIAMS-A. The scenario DRIAMS-A–B achieves the lowest AUROC scores between 0.3 and 0.35. It is noticeable that very high or low learning rates lead to performance decreases, per contra a low weight decay value, between 0.0 and 0.002, is favourable for the architecture. However, the model remains stable in AUROC scores within one scenario and we cannot make out one specific optimal hyperparameter set, as the model yields good performances for most parameter sets. Additionally, very low weight decay values can lead to overfitting, which remains a problem across all scenarios, as depicted in 6. In general, this plot and the loss curves suggest that the model is incapable of generalising from one domain to another as it does not learn the resistance mechanisms but overfits to the noise in the data.

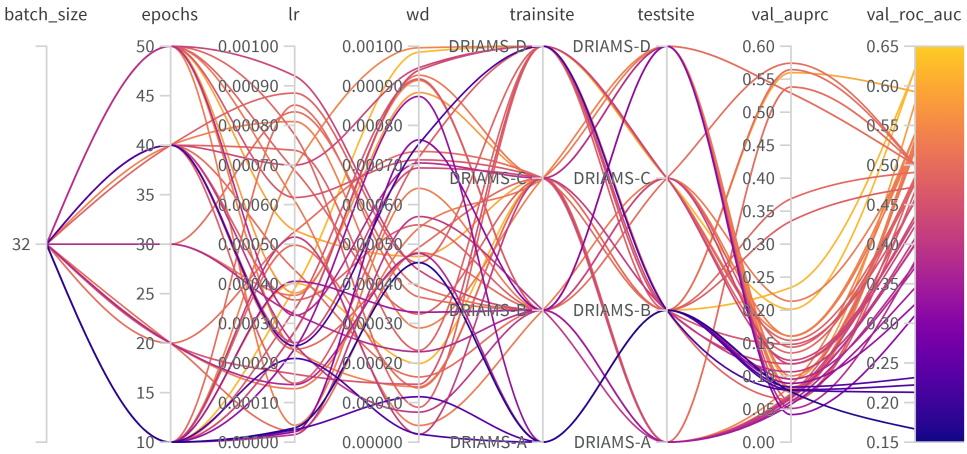


Figure 5.7: This parallel coordinate plot shows the results of the model exploration of the VGGNet classifier. The logged values of the runs for which parameter search and optimisation was carried out by W&B: Batch size, number of epochs, learning rate (lr), weight decay (wd), train dataset (trainsite), test dataset (testsite), AUPRC (val_auprc), AUROC (val_roc_auc). The best performances are reached in the scenarios DRIAMS-C–B and DRIAMS-D–C with AUROC scores of 0.62. The worst performing scenario, DRIAMS-A–B, reaches an AUROC score of 0.17, which is very low. Bigger learning rates, between 0.004 and 0.009, associate with better performances. The same is true for weight decay values between 0.0001 and 0.0004, suggesting that small regularisation is necessary. The model exhibits certain instability especially within the scenarios DRIAMS-D–C and DRIAMS-C–D, the AUROC scores differ by up to 0.2. This parameter sensitivity is typical for deep neural networks, however W&B is incapable of finding optimal parameters for our task. The model's generalisation performance is weak, as it performs worse than or similar to a random guess ($AUROC \leq 0.5$) in most cases.

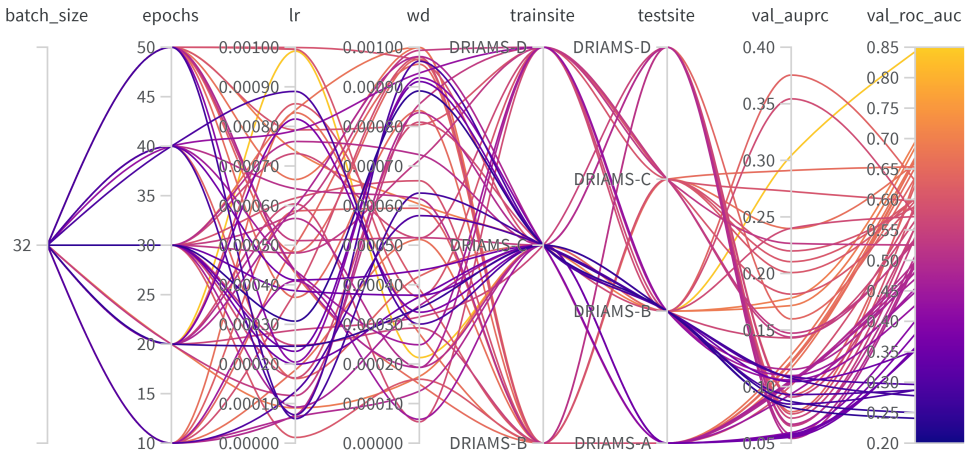


Figure 5.8: This parallel coordinate plot shows the results of the model exploration of the DenseNet classifier. The logged values of the runs for which parameter search and optimisation was carried out by W&B: Batch size, number of epochs, learning rate (lr), weight decay (wd), train dataset (trainsite), test dataset (testsite), AUPRC (val_auprc), AUROC (val_roc_auc). For this classifier, we leave out the scenario DRIAMS-A–B as the runtime surpasses 15 hours for one run within a sweep for a batch size of 200; a batch size of 32 would calculate even longer. The runtime is too long for the application in AMR prediction and the classifier is, therefore, negligible. The best performance is reached for the scenario DRIAMS-C–B with Escherichia coli and an AUROC score of 0.85. This is very high compared to other models. However, this run is an exception, as the runs from other scenarios accumulate between 0.7 and 0.4. The worst performance is also achieved in DRIAMS-C–B, but with Klebsiella pneumoniae with a score of 0.25. This highlights the strong dependency of the prediction outcome on the dataset. Clear trends in hyperparameters cannot be observed, nevertheless mid-range or higher learning rates and mid-range or lower weight decay values are oftentimes associated with higher AUROC scores. A small weight decay value helps the model learn complex patterns, but this does not succeed, as the AUROC values exhibit weak generalisation ability.

5.3.2 Baseline experiments: LeNet5, AlexNet

We conduct the described baseline experiment for all the classifiers mentioned in our Methods part 4.2. However, we were not able to generate extensive validation results for the Vgg16, Vgg19 and DenseNet classifiers, as the computational complexity of this task overwhelms our computing power. Hence, we further concentrate on the AlexNet and Lenet5 classifiers.

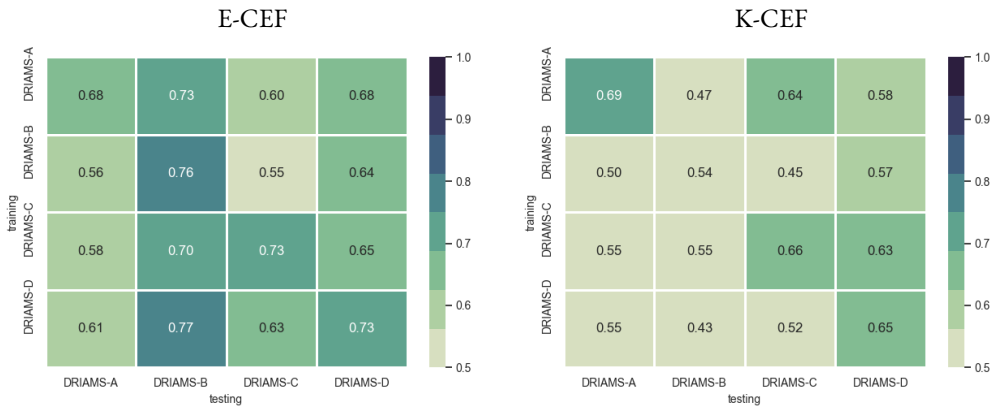
The LeNet5 classifier reaches a predictive performance similar to that of the MLP classifier for E-CEF 5.9a. Per contra, the classifier slightly underperforms for K-CEF 5.9b in comparison to the baseline AUROC values 5.1b. For K-CEF, the LeNet5 classifier and the MLP achieve a mean AUROC value of 0.56 and 0.58. For E-CEF, they reach mean AUROC values of 0.66 and 0.67, respectively. We also observe very similar mean AUPRC values for both species, more precisely the LeNet5 classifier achieves mean AUPRC scores 0.188 and 0.312 for K-CEF and E-CEF respectively, whereas the MLP classifier achieves 0.196 and 0.309. The LeNet5 classifier generates the best performance when trained on DRIAMS-D and DRIAMS-C and tested on a deviating site for E-CEF and K-CEF with mean AUROC values of 0.670 and 0.578. The test-sites DRIAMS-B and DRIAMS-D yield the best classification performances when trained on any other site. The scenario DRIAMS-D-B for E-CEF reaches the best generalisation performance for the LeNet5 classifier with an AUROC score of 0.77. However in general, when analysing the results and the corresponding loss curves, we find that the model overfits for most train-test scenarios.

The AlexNet classifier cannot quite compete with the baseline or the LeNet5 classifier 5.10: It achieves mean AUROC values of 0.55 and 0.62 for K-CEF and E-CEF; therefore, it underperforms by 0.03 and 0.05 compared to our baseline. Meanwhile, the mean AUPRC values reach 0.18 and 0.28 for K-CEF and E-CEF, deviating by 0.02 in both classifiers in comparison to the baseline.

The classifier exhibits the best generalisation performance for the train-sites DRIAMS-A and DRIAMS-C for E-CEF and K-CEF respectively, generating mean AUROC values of 0.65 and 0.58 when tested on the remaining sites. The test-site DRIAMS-D yields the best performances when trained on deviating sites in both cases, E-CEF and K-CEF. In a more thorough inspection, the classifier underperforms in scenarios tested on the DRIAMS-A dataset. Additionally, it only generates AUROC scores greater than 0.7 for scenarios trained and tested on the same site. Therefore, the AlexNet classifier exhibits a slightly weaker generalisation ability than the baseline.

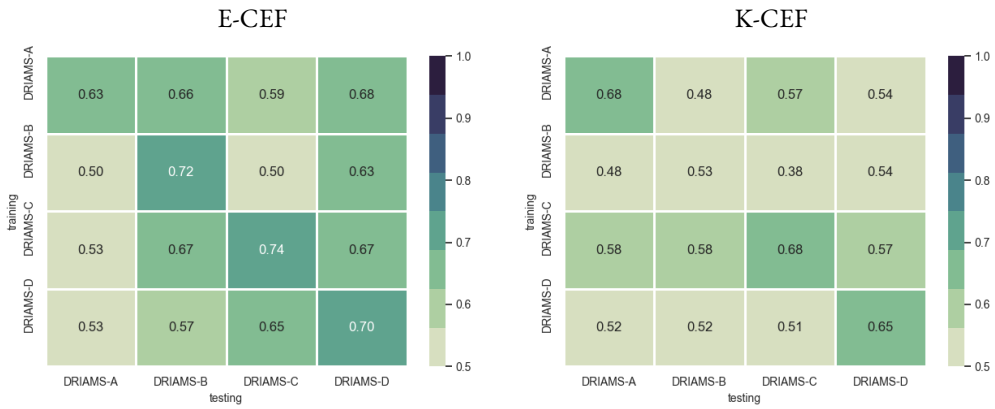
We observe a performance gap when predicting on spectra of the species *Klebsiella pneumonia* compared to spectra of the species *Escherichia coli*, a discrepancy of 0.1 and 0.07 in terms of AUROC can be found for the Lenet5 and AlexNet classifiers. Similarly, the baseline experiment for the MLP classifier also yields a performance gap of 0.08 in AUROC scores, showing that the deep neural networks do not exacerbate the gap. As mentioned above, this phenomenon can be attributed to a smaller dataset for the *Klebsiella pneumonia* species.

The baseline performance, as well as the model exploration, do not yield significant or any performance increases in terms of generalisation performance. Additionally, we find that overfitting remains a problem in most classifiers. To remedy this, we perform an ablation study.



(a) (b)

Figure 5.9: Mean AUROC performance of the same 10 random train-test splits using the LeNet5 classifier for combinations Escherichia coli-Ceftriaxone (left) and Klebsiella pneumoniae-Ceftriaxone (right). The classifier achieves a mean AUROC performance of 0.66 and 0.56 for E-CEF and K-CEF, respectively.



(a) (b)

Figure 5.10: Mean AUROC performance of the same 10 random train-test splits using the AlexNet classifier for combinations Escherichia coli-Ceftriaxone (left) and Klebsiella pneumoniae-Ceftriaxone (right). The classifier achieves a mean AUROC performance of 0.62 and 0.55 for E-CEF and K-CEF, respectively.

5.4 Ablated Neural Networks on Datasets from Multiple Domains

Contrary to our common experimental setup 5.1, we decrease the scope of the ablation experiments due to time constraints. We only use two domains: DRIAMS-A as the training site, DRIAMS-B as the test site. We choose these two domains, as they resemble real world scenarios more than other given scenarios: DRIAMS-A is a very big dataset, consisting of about 86.16 GB of data, whereas DRIAMS-B is a smaller dataset comprising 3.69 GB data. Furthermore, the classification performance of the MLP in *maldi_amr* lacks in this scenario compared to other train-test site combinations.

In these experiments we execute training, testing and validation for the same seven different random train-test splits to decrease performance variation due to the specific split. However, we do not apply cross-validation nor early stopping however, we do apply z-score normalisation. We plot the training- and the validation loss and assess model performance using the mean AUROC, mean AUPRC and mean accuracy of all train-test splits of one classifier architecture.

We find that the initial LeNet5 model architecture with two convolutional and three fully connected layers performs worse than models with fewer layers 5.11. The classifier comprised of two convolutional layers and one fully connected layer (*lenet_012*) achieves an AUROC score of 0.72, outperforming the initial architecture by 0.09. Furthermore, the classifier comprised of one convolutional layer and three fully connected layers *(lenet_0234*) outperforms the initial architecture by 0.07 in terms of AUPRC values. We analyse the generated loss curves and find that loss curves of *lenet_0234* converge better than those of *lenet_012*. The learning curves of the different architectures mainly indicate underfitting, but for *lenet_0234*, we can observe loss curves that converge more stably in comparison. Therefore, we choose the best-performing classifier in terms of AUPRC, *lenet_0234*.

The ablation study for the AlexNet architecture renders more straightforward results: We find that the AUPRC values mostly increase with a decreasing number of convolutional layers. Additionally, the AUROC values increase with an increasing number of fully connected layers. However, an exception occurs for the architecture consisting of two fully connected layers and three convolutional layers, having the highest mean AUROC score of 0.67. The validation and training loss converge in every random train-test seed 5.13, the learning curves of the *alex_01256* indicate that the model is able to generalise to unseen data. Consequently, we choose this model architecture for further validation, *alex_01256*.

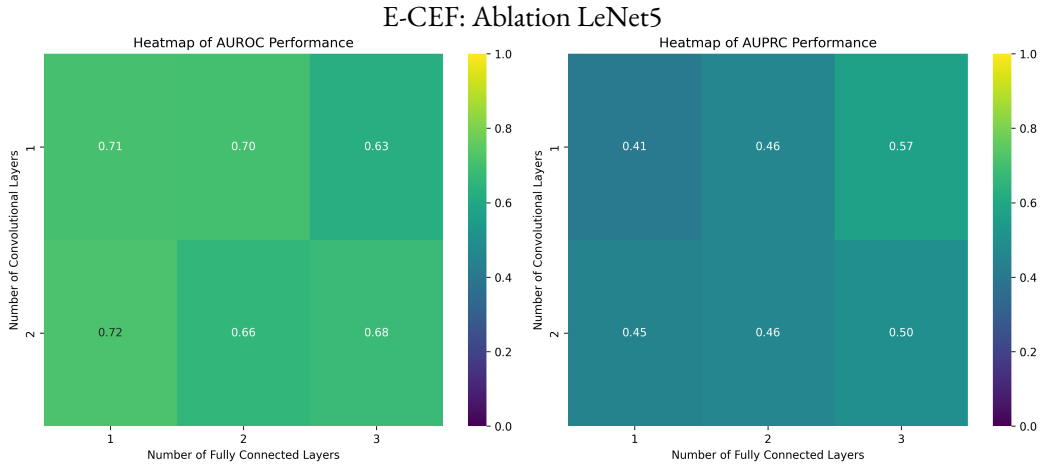


Figure 5.11: This figure shows the mean AUROC and AUPRC performance of 6 different model architectures of the same seven random train-test splits. The x-axis shows the number of convolutional layers, the y-axis shows the number of fully connected layers within one architecture. The best AUROC score is achieved by the model architecture with two convolutional layers and one fully connected layer; the best AUPRC score is achieved by the model comprising one convolutional layer and three fully connected layers.

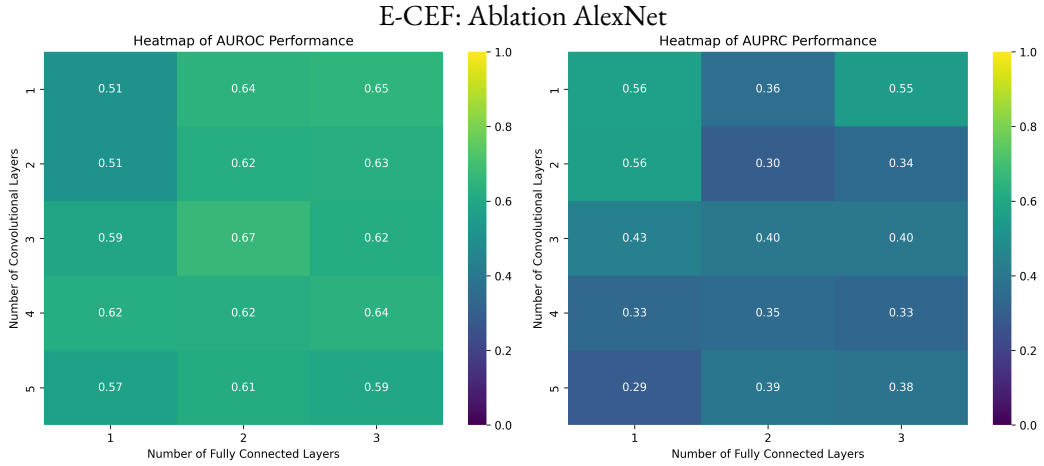


Figure 5.12: This figure shows the AUROC and AUPRC performance of 15 different model architectures of the same seven random train-test splits. The x-axis shows the number of convolutional layers, the y-axis shows the number of fully connected layers within one architecture. The best AUROC score is achieved by the model architecture with three convolutional layers and two fully connected layers; the best AUPRC score is achieved by the models comprising one or two convolutional layers and one fully connected layer.

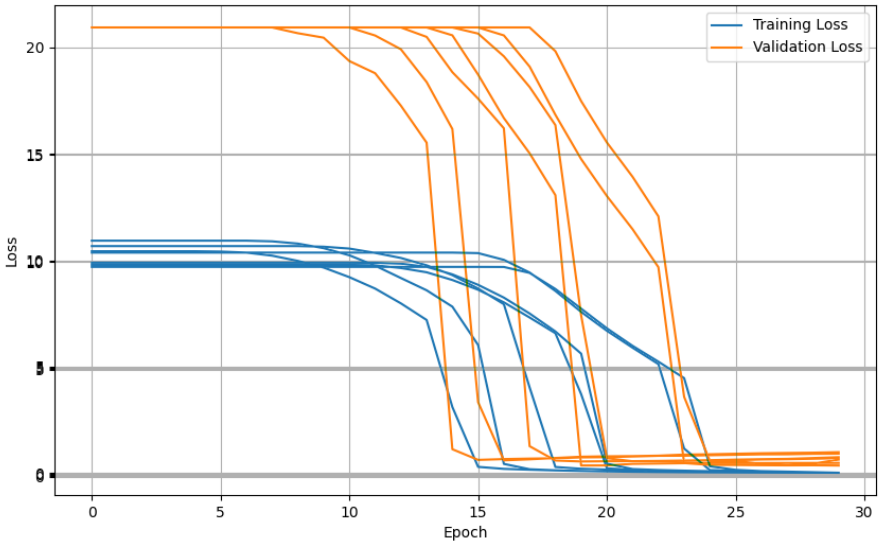


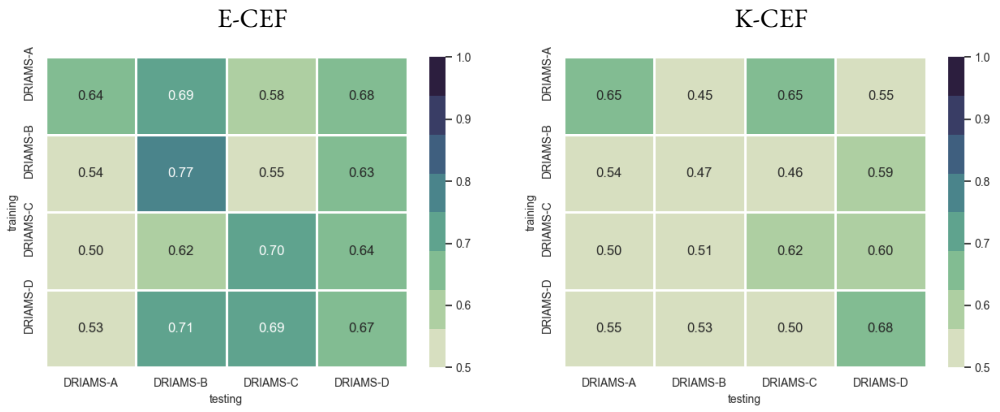
Figure 5.13: This plot depicts the training and validation loss curves for seven different train-test splits for the *alex_01256* classifier. The training loss is depicted in blue; the validation loss is marked in orange. As opposed to other model architectures, the classifier achieves very similar loss curves across the seven different train-test splits. The training- and validation losses converge in tune for every split, meaning that the losses decrease at similar epochs and with similar extremes.

5.4.1 Baseline experiment: modified networks

We conduct the described baseline experiment for the chosen classifiers: *lenet_0234* and *alex_01256*.

We find that the chosen *lenet_0234* model does not render performance increases. It yields mean AUROC performances of 0.55 and 0.63 for K-CEF and E-CEF, underperforming by 0.01 and 0.02 compared to the original LeNet5 model. Furthermore, the ablated model deviates in the way that the best-performing train-sites are DRIAMS-A for both species as opposed to DRIAMS-D and DRIAMS-C in the basic LeNet5 architecture. Additionally, the best generalisation performance, meaning scenarios with deviating train-test sites, is achieved for E-CEF for the DRIAMS-D–B scenario with a score of 0.71.

These deviations can occur because we only ablate using the train-test scenario DRIAMS-A–DRIAMS-B and have to use a smaller grid for our grid search functionality, due to timely and computational limits. Therefore, we risk that our model architecture will only perform well in this scenario, but other scenarios will necessitate deeper architectures. However, we find that the discrepancies in performance are minor, underlining that the chosen architecture does not significantly affect the validation results of our models, nor does it affect the generalisation performance.



(a) (b)

Figure 5.14: Mean AUROC performance of the same 10 random train-test splits using the *lenet_0234* classifier for combinations Escherichia coli-Ceftriaxone (left) and Klebsiella pneumoniae-Ceftriaxone (right). The classifier reaches mean AUROC values of 0.55 and 0.63 for K-CEF and E-CEF, respectively.

6 Conclusion

We extend prior work on antimicrobial resistance prediction with the goal of increasing classifier generalisation performance. We do this by training and evaluating three different deep artificial neural networks. We conduct a multitude of experiments on 16 different train-test scenarios using the AlexNet and LeNet5 model architectures. We arrive at the conclusion that deep classifiers and parameter tuning, to our extent, alone are not sufficient for achieving the desired domain adaptation and consequently do not increase generalisation performance significantly. To further examine model behaviour, we introduce an ablation study over the layers of the LeNet5 and AlexNet model architectures.

In this thesis, we show that deep neural networks, especially CNNs, can compete with other machine learning methods in the realm of antimicrobial resistance prediction. However, we recognise that we have not found the optimal parameters or network architecture for realising better generalisation performance compared to other publications.

Things we tried but did not work In the course of this work, we tried various approaches, which we had to dismiss due to unfavourable results. To contribute to future work, we want to name methodologies that failed to increase performance in our experiments. As we make clear, we try various prominent model architectures and optimise model parameters using W&B. However, we do not observe stark differences in performance, only in computing time. A computing time of more than five hours for a single training, testing and validation process is why we exclude the ResNet18, ResNet50, and VggNet19 architectures from this thesis.

As we constantly face the problem of overfitting, we also include dropout and attention mechanisms, especially in the LeNet5 architecture. Nevertheless, the issue of model instability remains, which is why we also implement the optuna hyperparameter optimisation framework but fail to generate helpful results. This is because we can only optimise for one train-test scenario, seed and antibiotic-bacterium combination. Furthermore, we also tune learning rates using the PyTorch learning rate finder and linear, polynomial and exponential learning rate schedulers. Even so, we cannot remedy over- or underfitting in most cases.

Final Implications This is where future contributions tie in; there are possibilities galore to further explore the scope of deep neural networks and the DRIAMS dataset. We have set the basis for transfer learning using CNNs as the base model; therefore exploring domain adaptation methods such as deep domain confusion (DDC), deep CORAL (Correlation Alignment), or deep adaptation networks (DAN) would be a logical next step. To acquire the optimal CNN model, one could extend the given ablation study to validate the results more thoroughly. For example, including all train-test scenarios (DRIAMS-A–DRIAMS-D), all ten random stratified train-test-splits, cross-validation, drop-out, regularisation methods and learning rate schedulers.

In analysing previous contributions and our work, we cannot observe significant differences in the performance of the presented classifiers, even though we use a variety of deep learning architectures of different sizes.

This leads us to the hypothesis that not only the classifiers need to be optimised but, more importantly, the data. Machine learning models are designed to find patterns or underlying information within the data. This task is exceptionally challenging in the application of AMR predictions because the occurring resistance mechanisms are not fully known to science [22]. Consequently, we are not aware of which features are crucial within our spectral data. As the data is preprocessed and binned in a specific way, potentially significant features within the data may be blurred or cancelled out. Hence, another critical aspect to inspect is the characteristics of the DRIAMS dataset and the final binned data we feed into our models. These could be the scale and representativeness of the data, particularly since grave performance gaps occur for varying train-test data splits. Lastly, above all, the binning process manipulates the intricacies of the spectra; therefore, experiments for optimising the spectral bin size could fill essential gaps in applying machine learning methods for AMR predictions.

Bibliography

1. T. Ali, S. Ahmed, and M. Aslam. "Artificial intelligence for antimicrobial resistance prediction: challenges and opportunities towards practical implementation". *Antibiotics* 12:3, 2023, p. 523.
2. D. Arora, M. Garg, and M. Gupta. "Diving deep in Deep Convolutional Neural Network". In: *2020 2nd International Conference on Advances in Computing, Communication Control and Networking (ICACCCN)*. 2020, pp. 749–751. DOI: 10.1109/ICACCCN51052.2020.9362907.
3. W. Bank. "Drug-resistant infections: a threat to our economic future", 2017.
4. A. P. Bradley. "The use of the area under the ROC curve in the evaluation of machine learning algorithms". *Pattern recognition* 30:7, 1997, pp. 1145–1159.
5. A. J. Browne, M. G. Chipeta, G. Haines-Woodhouse, E. P. Kumaran, B. H. K. Hamadani, S. Zaraa, N. J. Henry, A. Deshpande, R. C. Reiner, N. P. Day, et al. "Global antibiotic consumption and usage in humans, 2000–18: a spatial modelling study". *The Lancet Planetary Health* 5:12, 2021, e893–e904.
6. R. M. Cichy and D. Kaiser. "Deep neural networks as scientific models". *Trends in cognitive sciences* 23:4, 2019, pp. 305–317.
7. A. Croxatto, G. Prod'hom, and G. Greub. "Applications of MALDI-TOF mass spectrometry in clinical diagnostic microbiology". *FEMS microbiology reviews* 36:2, 2012, pp. 380–407.
8. K. documentation. *Model training APIs*. https://keras.io/api/models/model_training_apis/.
9. M. Feucherolles, M. Nennig, S. L. Becker, and C. Ragimbeau. "Combination of MALDI-TOF mass spectrometry and machine learning for rapid antimicrobial resistance screening: The case of Campylobacter spp." *Frontiers in Microbiology* 12, 2022, p. 804484.
10. H. Fırat, M. E. Asker, M. İ. Bayindir, and D. Hanbay. "Spatial-spectral classification of hyperspectral remote sensing images using 3D CNN based LeNet-5 architecture". *Infrared Physics & Technology* 127, 2022, p. 104470.
11. J. Fu, F. He, J. Xiao, Z. Liao, L. He, J. He, J. Guo, and S. Liu. "Rapid AMR prediction in Pseudomonas aeruginosa combining MALDI–TOF MS with DNN model". *Journal of Applied Microbiology* 134:11, 2023, lxad248.
12. G. Huang, Z. Liu, L. Van Der Maaten, and K. Q. Weinberger. "Densely connected convolutional networks", 2017, pp. 4700–4708.
13. M. Irfan, A. Almotiri, and Z. A. AlZeyadi. "Antimicrobial resistance and its drivers—A review". *Antibiotics* 11:10, 2022, p. 1362.

14. O. Jonas, A. Irwin, F. Berthe, F. Le Gall, and P. Marquez. "Drug-resistant infections: a threat to our economic future (vol. 2): final report (English). The World Bank", 2019.

15. A. Krizhevsky, I. Sutskever, and G. E. Hinton. "ImageNet classification with deep convolutional neural networks". *Communications of the ACM* 60:6, 2017, pp. 84–90.

16. A. Krogh. "What are artificial neural networks?" *Nature biotechnology* 26:2, 2008, pp. 195–197.

17. S. Kumar. "Antimicrobial resistance: A top ten global public health threat". *EClinicalMedicine* 41, 2021, p. 101221.

18. D. Li, J. Yi, G. Han, and L. Qiao. "MALDI-TOF Mass Spectrometry in Clinical Analysis and Research". *ACS Measurement Science Au* 2:5, 2022, pp. 385–404. DOI: 10.1021/acsmeasuresciau.2c00019.

19. S. Madhavan. "Introduction to convolutional neural networks", 2021. Access Date: April 17, 2024. URL: https://developer.ibm.com/articles/introduction-to-convolutional-neural-networks/.

20. R. Meyes, M. Lu, C. de Puiseau, and T. Meisen. "Ablation studies in artificial neural networks. arXiv 2019". *arXiv preprint arXiv:1901.08644*, 1901.

21. P. Mishra. "Introduction to PyTorch, Tensors, and Tensor Operations". In: *PyTorch Recipes: A Problem-Solution Approach to Build, Train and Deploy Neural Network Models*. Apress, Berkeley, CA, 2023, pp. 1–28. ISBN: 978-1-4842-8925-9. DOI: 10.1007/978-1-4842-8925-9_1.

22. J. M. Munita and C. A. Arias. "Mechanisms of Antibiotic Resistance". *Microbiology Spectrum* 4:2, 2016, 10.1128/microbiolspec.vmbf-0016–2015. DOI: 10.1128/microbiolspec.vmbf-0016-2015.

23. C. J. Murray, K. S. Ikuta, F. Sharara, L. Swetschinski, G. R. Aguilar, A. Gray, C. Han, C. Bisignano, P. Rao, E. Wool, et al. "Global burden of bacterial antimicrobial resistance in 2019: a systematic analysis". *The lancet* 399:10325, 2022, pp. 629–655.

24. J. O'Neill. "Tackling drug-resistant infections globally: final report and recommendations", 2016.

25. I. Ofeidis, D. Kiedanski, and L. Tassiulas. *An overview of the data-loader landscape: Comparative performance analysis*. 2022.

26. G. W. H. Organisation. "Global antimicrobial resistance and use surveillance system (GLASS) report 2022", 2022. URL: https://iris.who.int/bitstream/handle/10665/364996/9789240062702-eng.pdf?sequence=1.

27. V. M. Pierce and A. J. Mathers. "Setting antimicrobial susceptibility testing breakpoints: a primer for pediatric infectious diseases specialists on the Clinical and Laboratory Standards Institute approach". *Journal of the Pediatric Infectious Diseases Society* 11:2, 2022, pp. 73–80.

28. PyTorch. *Datasets DataLoaders*. https://pytorch.org/tutorials/beginner/basics/data_tutorial.html. Accessed: 2024-03-13. 2024.

29. PyTorch. *Introduction to PyTorch Tensors*. Accessed: 2024-02-26. 2024.

30. Y. Ren, T. Chakraborty, S. Doijad, L. Falgenhauer, J. Falgenhauer, A. Goesmann, O. Schwengers, and D. Heider. "Deep transfer learning enables robust prediction of antimicrobial resistance for novel antibiotics". *Antibiotics* 11:11, 2022, p. 1611.

31. K. Simonyan and A. Zisserman. "Very deep convolutional networks for large-scale image recognition". *arXiv preprint arXiv:1409.1556*, 2014.

32. N. Tarfeen, K. U. Nisa, and Q. Nisa. "MALDI-TOF MS: application in diagnosis, dereplication, biomolecule profiling and microbial ecology". *Proceedings of the Indian National Science Academy* 88:3, 2022, pp. 277–291.

33. G. Visonà, D. Duroux, L. Miranda, E. Sükei, Y. Li, K. Borgwardt, and C. Oliver. "Multimodal learning in clinical proteomics: enhancing antimicrobial resistance prediction models with chemical information". *Bioinformatics* 39:12, 2023, btad717.

34. H. Wallen. "Mass spectrometry using electrospray ionization". Version 3. *Nature Reviews Methods Primers* 1, 2023, p. 24. DOI: 10.1038/s43586-023-00219-w.

35. H.-Y. Wang, C.-R. Chung, Z. Wang, S. Li, B.-Y. Chu, J.-T. Horng, J.-J. Lu, and T.-Y. Lee. "A large-scale investigation and identification of methicillin-resistant Staphylococcus aureus based on peaks binning of matrix-assisted laser desorption ionization-time of flight MS spectra". *Briefings in Bioinformatics* 22:3, 2021, bbaa138.

36. H.-Y. Wang, T.-T. Hsieh, C.-R. Chung, H.-C. Chang, J.-T. Horng, J.-J. Lu, and J.-H. Huang. "Efficiently predicting vancomycin resistance of Enterococcus faecium from MALDI-TOF MS spectra using a deep learning-based approach". *Frontiers in Microbiology* 13, 2022, p. 821233.

37. C. Weis, A. Cuénod, B. Rieck, K. Borgwardt, and A. Egli. *DRIAMS: database of resistance information on antimicrobials and MALDI-TOF mass spectra*. 2021.

38. 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 Medicine* 28:1, 2022, pp. 164–174.

39. C. Weis, M. Horn, B. Rieck, A. Cuénod, A. Egli, and K. Borgwardt. "Topological and kernel-based microbial phenotype prediction from MALDI-TOF mass spectra". *Bioinformatics* 36:Supplement_1, 2020, pp. i30–i38.

40. C. Weis, B. Rieck, S. Balzer, A. Cuénod, A. Egli, and K. Borgwardt. "Improved MALDI-TOF MS based antimicrobial resistance prediction through hierarchical stratification", 2022. DOI: 10.1101/2022.04.13.488198.

41. C. V. Weis, C. R. Jutzeler, and K. Borgwardt. "Machine learning for microbial identification and antimicrobial susceptibility testing on MALDI-TOF mass spectra: a systematic review". *Clinical Microbiology and Infection* 26:10, 2020, pp. 1310–1317.

42. C. V. Weis. "MALDI-TOF mass spectrometry based clinical antimicrobial resistance prediction using machine learning". Doctoral Thesis. Zurich: ETH Zürich, 2022. DOI: 10.3929/ethz-b-000535463.

43. P. Werbos. “Backpropagation through time: what it does and how to do it”. *Proceedings of the IEEE* 78:10, 1990, pp. 1550–1560. DOI: 10.1109/5.58337.

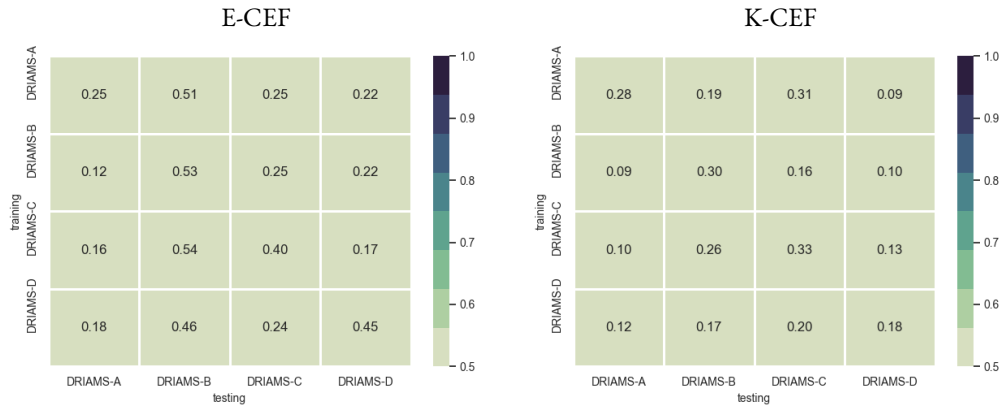
44. J. Yu, Y.-T. Lin, W.-C. Chen, K.-H. Tseng, H.-H. Lin, N. Tien, C.-F. Cho, J.-Y. Huang, S.-J. Liang, L.-C. Ho, et al. “Direct prediction of carbapenem-resistant, carbapenemase-producing, and colistin-resistant Klebsiella pneumoniae isolates from routine MALDI-TOF mass spectra using machine learning and outcome evaluation”. *International Journal of Antimicrobial Agents* 61:6, 2023, p. 106799.

45. Y.-M. Zhang, M.-F. Tsao, C.-Y. Chang, K.-T. Lin, J. J. Keller, and H.-C. Lin. “Rapid identification of carbapenem-resistant Klebsiella pneumoniae based on matrix-assisted laser desorption ionization time-of-flight mass spectrometry and an artificial neural network model”. *Journal of Biomedical Science* 30:1, 2023, p. 25.

Appendices

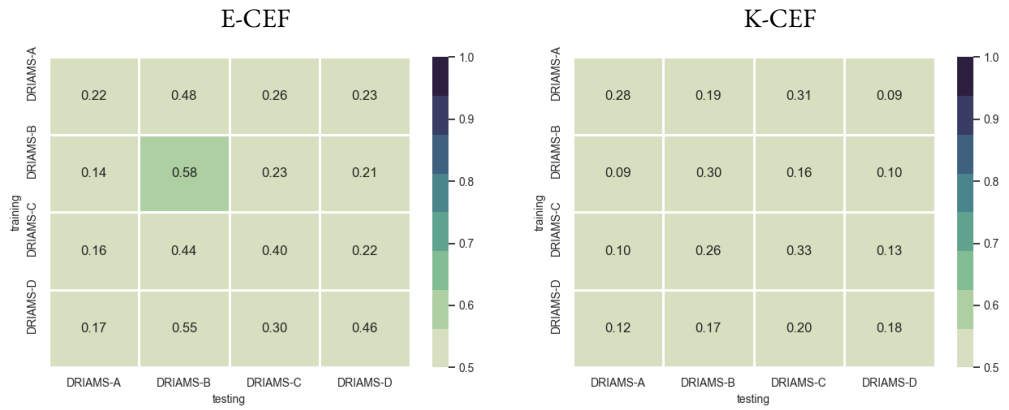
Additional Figures

In the scope of this thesis, we focus on the AUROC metric for comparability to prior work. However, the AUPRC metric is equally important for generalisation tasks. Therefore, we want to present the classifiers' AUPRC performances more elaborately.



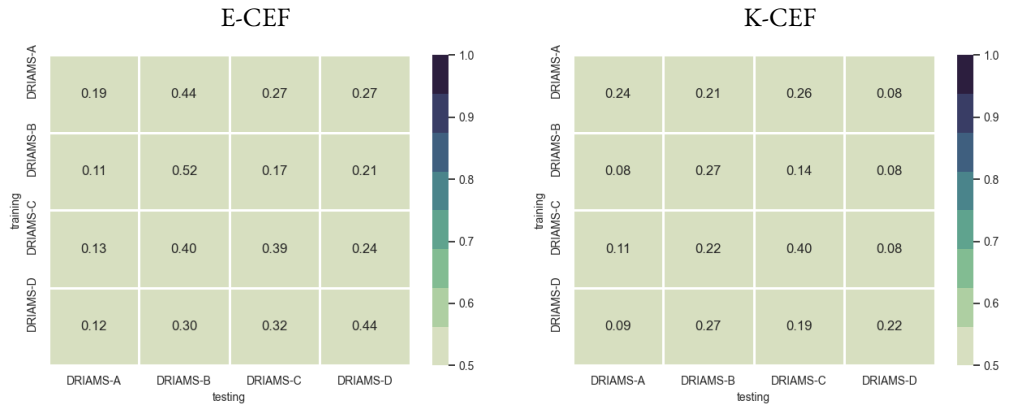
(a) (b)

Figure 1: Mean AUPRC performance of the same 10 random train-test splits using the MLP classifier implemented using PyTorch. The classifier achieves mean AUPRC scores of 0.309 and 0.186 for E-CEF and K-CEF, respectively. The best generalisation performances of 0.31 and 0.54 are achieved in the scenarios DRIAMS-A–C and DRIAMS-C–B for K-CEF and E-CEF. The AUPRC performance is increased for test-sites DRIAMS-B and DRIAMS-C for both species.



(a) (b)

Figure 2: Mean AUPRC performance of the same 10 random train-test splits using the LeNet5 classifier. The classifier achieves mean AUPRC scores of 0.312 and 0.188 for E-CEF and K-CEF, respectively. The best generalisation performances of 0.31 and 0.55 are achieved in the scenarios DRIAMS-A–C and DRIAMS-D–B for K-CEF and E-CEF. The AUPRC performance is increased for test-sites DRIAMS-B and DRIAMS-C for both species.



(a) (b)

Figure 3: Mean AUPRC performance of the same 10 random train-test splits using the AlexNet classifier. The classifier achieves mean AUPRC scores of 0.28 and 0.18 for E-CEF and K-CEF, respectively. The best generalisation performances of 0.27 and 0.44 are achieved in the scenarios DRIAMS-D–B and DRIAMS-A–B for K-CEF and E-CEF. The AUPRC performance is increased for test-sites DRIAMS-B and DRIAMS-C for both species.

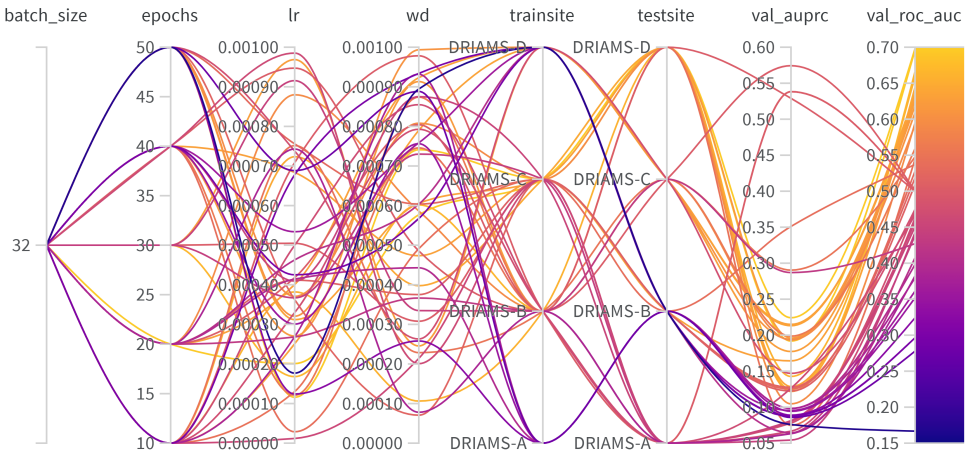


Figure 4: This parallel coordinate plot shows the results of the model exploration of the ResNet18 classifier, only for the species *Klebsiella pneumoniae*. The logged values of the runs for which parameter search and optimisation was carried out by W&B: Batch size, number of epochs, learning rate (lr), weight decay (wd), train dataset (trainsite), test dataset (testsite), AUPRC (val_auprc), AUROC (val_roc_auc). We leave this model out from the main scope of the thesis, as the computational complexity overwhelms our system. The classifier predicts AUROC values in the range of [0.3; 0.7]. The best performance is achieved in scenarios tested on DRIAMS-D or DRIAMS-C. The whole range of the hyperparameter is used and no clear trends can be observed; however, we find that mid-range to lower weight decay values are oftentimes associated with better performance.

Furthermore, we want to exhibit the loss curves from the report, which we reference in 5.3.1. The following figures show two plots. The top plot shows the validation loss curves for each run of each sweep within the model exploration experiment. The bottom plot shows the training loss curves for the same runs of the same sweeps; the corresponding loss- and validation curves are plotted using the same colour and pattern. For a better overview, go to the mentioned report to be able to scroll and select individual runs.

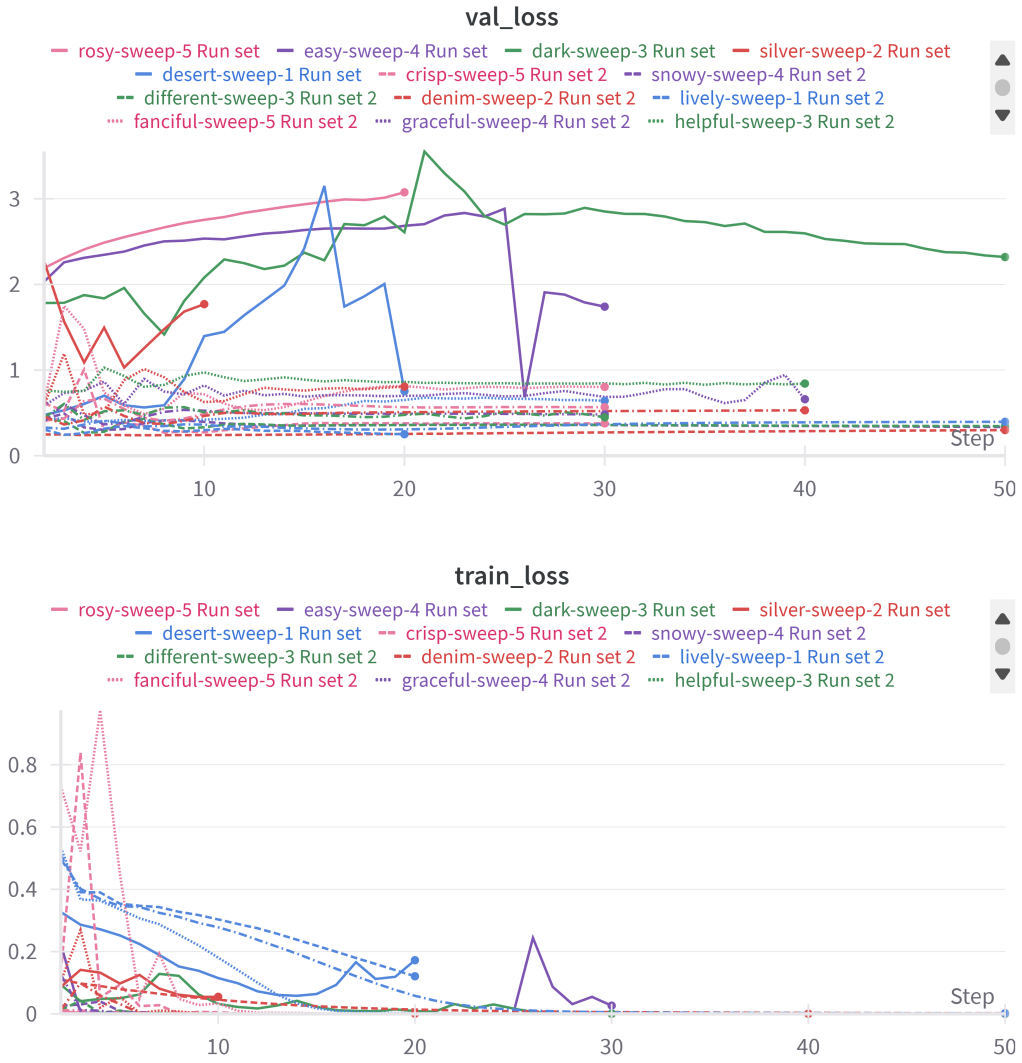


Figure 5: This figure shows the plots for the validation- (val_loss) and training (train_loss) loss curves for the PyTorch-MLP classifier. The training loss curves continuously decrease and converge to zero in most cases, suggesting that the model learns the data well. The validation loss curves do not converge; the curves either remain stable without change, suggesting underfitting, or increase by growing epochs and jump between values, suggesting overfitting. However, this indicates that the model cannot learn the underlying resistance patterns.

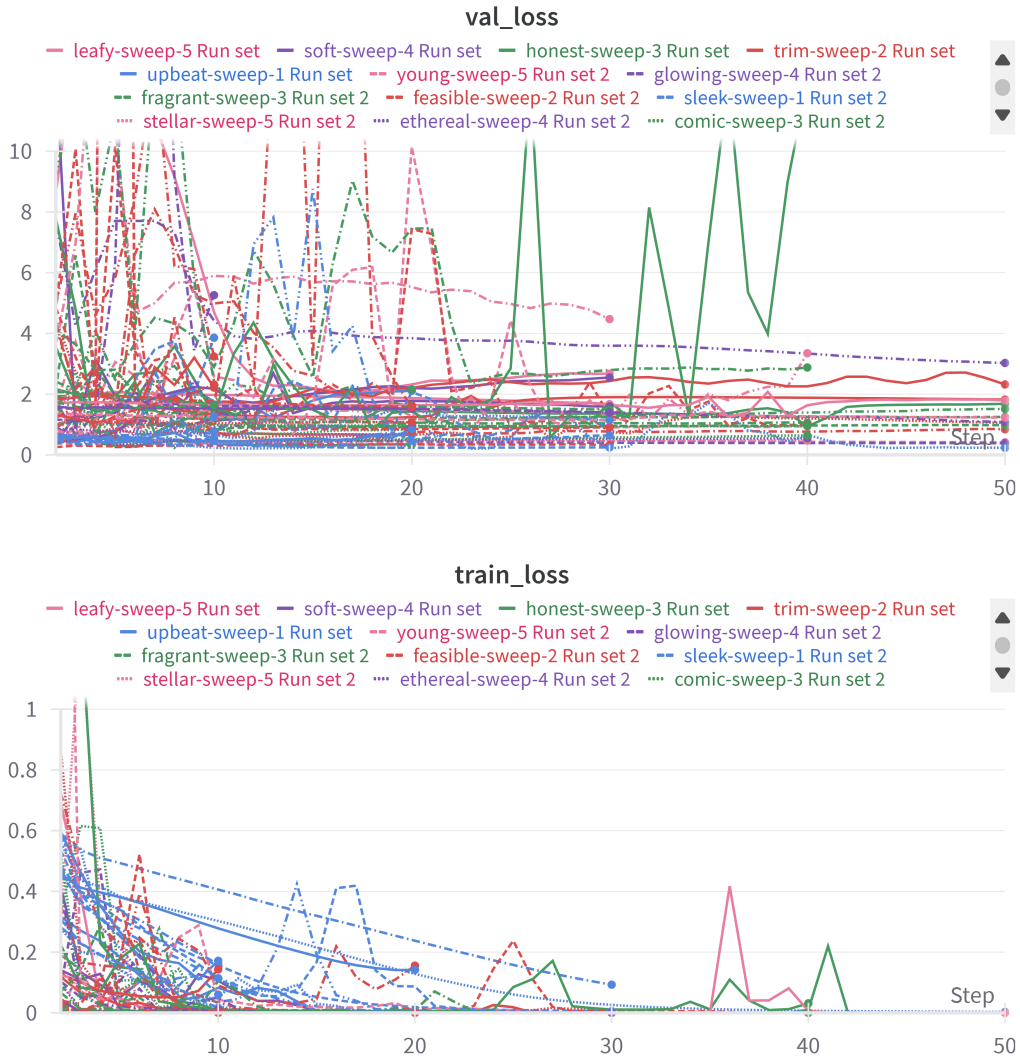


Figure 6: This figure shows the plots for the validation- (val_loss) and training (train_loss) loss curves for the LeNet5 classifier. A number of training loss curves converge smoothly. However, many of them fluctuate between values, which suggests that the model parameters, like the learning rate, are set inadequately or that the data contains too much noise. The corresponding validation loss curves show similar behaviour: the curves either remain stable without converging, indicating underfitting, or increase by growing epochs and jump between values, indicating overfitting. Possible causes could be that the model cannot generalise to the validation data due to inadequate model complexity or unrepresentative data.

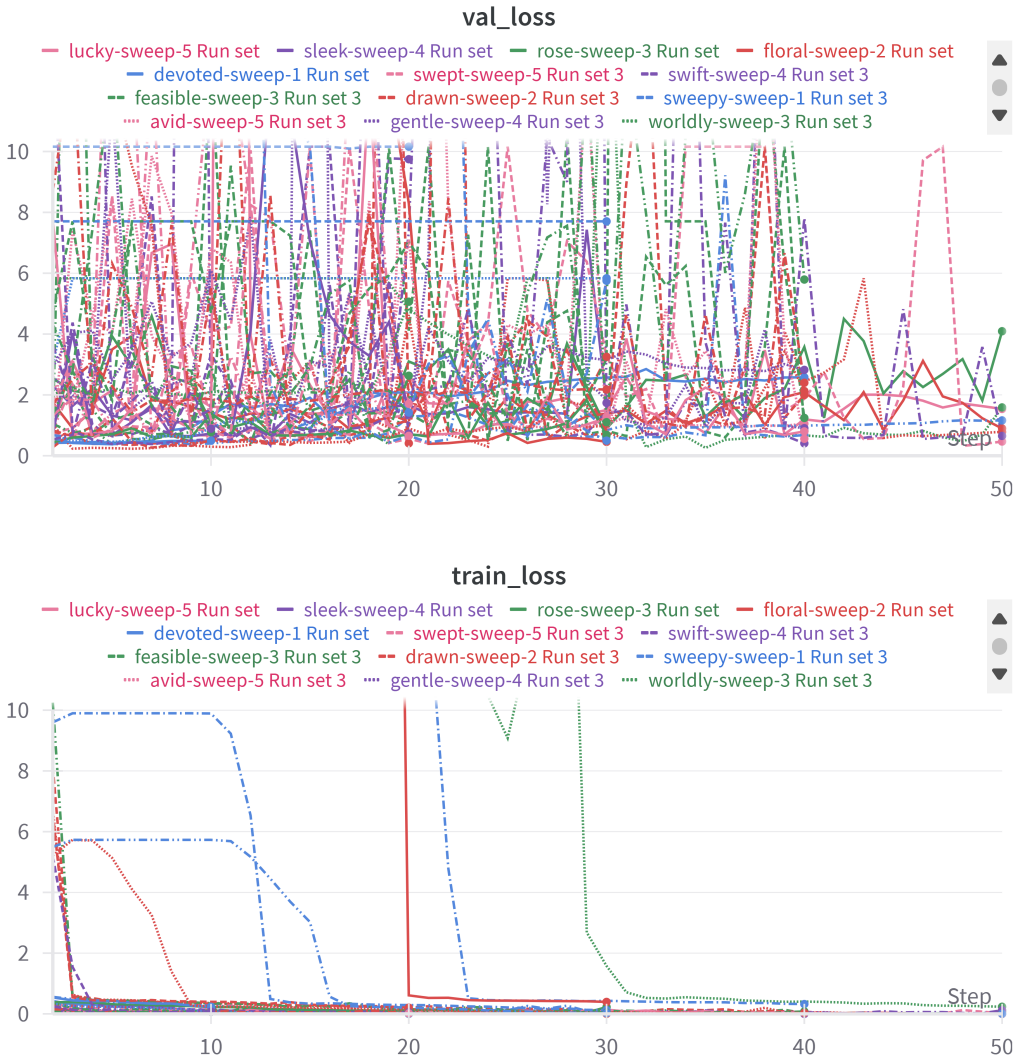


Figure 7: This figure shows the plots for the validation- (val_loss) and training (train_loss) loss curves for the AlexNet classifier. The training loss curves consistently converge to the value zero; either very slowly or with an extreme drop, indicating that the learning rate is either too high or too low. Conversely, the validation loss curves exhibit extreme jumps between values. The model is highly inconsistent and cannot generalise to the validation data, which can be caused by overfitting.