CHALMERS

Predicting antibiotic resistance using fusion transformers

A framework for training a multimodal transformer using data fusion of genotype, phenotype, and metadata to improve predictions of antibiotic resistance in *Escherichia coli*

Master's thesis in Complex Adaptive Systems

Jesper Olsson

DEPARTMENT OF MATHEMATICAL SCIENCES

CHALMERS UNIVERSITY OF TECHNOLOGY
Gothenburg, Sweden 2024
www.chalmers.se

Master's thesis 2024

Predicting antibiotic resistance using fusion transformers

A framework for training a multimodal transformer using data fusion of genotype, phenotype, and metadata to improve predictions of antibiotic resistance in *Escherichia coli*

JESPER OLSSON

CHALMERS
UNIVERSITY OF TECHNOLOGY

Department of Mathematical Sciences
Division of Applied Mathematics and Statistics
Erik Kristiansson group
Chalmers University of Technology
Gothenburg, Sweden 2024

Predicting antibiotic resistance using fusion transformers
A framework for training a multimodal transformer using data fusion of genotype, phenotype, and metadata to improve predictions of antibiotic resistance in *Escherichia coli*
JESPER OLSSON

Supervisors:
Anna Johnning, Department of Mathematical Sciences at Chalmers University of Technology and University of Gothenburg & Department of Systems and Data Analysis at Fraunhofer-Chalmers Centre
Erik Kristiansson, Department of Mathematical Sciences
Examiner: Erik Kristiansson, Department of Mathematical Sciences

Master's Thesis 2024
Department of Mathematical Sciences
Division of Applied Mathematics and Statistics
Erik Kristiansson group
Chalmers University of Technology
SE-412 96 Gothenburg
Telephone +46 31 772 1000

Typeset in LaTeX
Printed by Chalmers Reproservice
Gothenburg, Sweden 2024

Predicting antibiotic resistance using fusion transformers
A framework for training a multimodal transformer using data fusion of genotype, phenotype, and metadata to improve predictions of antibiotic resistance in *Escherichia coli*

JESPER OLSSON
Department of Mathematical Sciences
Chalmers University of Technology

Abstract

Antimicrobial resistance threatens recent gains in global public health by making it more difficult to treat infections. Clinicians must administer treatments based on limited diagnostic information and increasing resistance complicates these decisions. This thesis project explores ways to support this process by developing a framework for training a transformer model using data fusion of patient and genotype data with phenotype data to make individualized predictions of antibiotic resistance in *Escherichia coli* on these multimodal data. To achieve this, the model was trained in two stages: first, the model was pre-trained on large volumes of unimodal data using masked language modeling to learn patterns within the modalities; and second, the model was fine-tuned on a small multimodal dataset to learn patterns across modalities. To evaluate pre-training strategies, the model was fine-tuned on two clinically relevant tasks and smaller training sets. To determine the value of introducing multimodality and the effect of genotype data availability on performance, the model was fine-tuned on varying levels of available genotype information.

The results show that the model performs well on the fine-tuning tasks, that pre-training on unimodal data improves performance, and that the model can extrapolate well from small training sets and incomplete data. Therefore, it can be concluded that this work has achieved the aim of developing a model that can make accurate predictions based on limited diagnostic information. Importantly, large performance improvements were observed with increasing genotype data availability, especially on difficult antibiotics. Furthermore, the model was better able to utilize available genotype information when pre-trained. However, while no clear conclusion on the best pre-training strategy can be drawn from the results of this work, they indicate that using systematic class masking in pre-training yields the highest performance. Future research should further investigate the best strategy for pre-training the model, how the model utilizes genotype data to improve performance, and how genotype data affects performance on limited training data.

Keywords: transformer, multimodal, machine learning, deep learning, data fusion, antibiotic resistance, artificial intelligence, masked language modeling, neural networks.

Acknowledgements

This thesis project has been an incredible opportunity and a pleasure. I owe it to my supervisors Erik Kristiansson and Anna Johnning, who have been insightful, patient, and supportive. I cannot count the number of times I've entered meetings with you in a state of disarray and worry and left excited, confident, and happy. Thank you for setting excellent examples of leadership that I will carry with me henceforth. Thank you, Erik, for your engagement, contagious curiosity, and insightful questions. Thank you, Anna, for your warm welcome, fresh perspectives, and excellent insights into the data. I would also like to thank everyone in Erik Kristiansson's group for their warm welcome and lovely social events.

I'd like to thank Lennart Svensson and David Hagerman Olzon for helping me get started with the framework and for their helpful tips on implementation. Furthermore, I thank my thesis colleagues, Michaela Holmström and Erik Aerts, for their friendship and rewarding discussions. With emphasis, I thank my family for their continuous support and my lovely partner, Emma, for your support and patience, and for believing in me even when I do not. Last but not least, I want to thank our dog, Apollo, for his fluffiness, his pleasant presence and patience during long sessions of coding and problem-solving, and for eventually forcing me to take well-needed breaks.

This thesis project utilizes data from the European Surveillance System - TESSy, provided by Andorra, Albania, Armenia, Austria, Azerbaijan, Bosnia and Herzegovina, Belgium, Bulgaria, Belarus, Switzerland, Cyprus, Czechia, Germany, Denmark, Estonia, Greece, Spain, Finland, France, Georgia, Croatia, Hungary, Ireland, Israel, Iceland, Italy, Kyrgyzstan, Kazakhstan, Liechtenstein, Lithuania, Luxembourg, Latvia, Monaco, Republic of Moldova, Montenegro, Republic of North Macedonia, Malta, Netherlands, Norway, Poland, Portugal, Romania, Serbia, Russian Federation, Sweden, Slovenia, Slovakia, San Marino, Tajikistan, Turkmenistan, Turkey, Ukraine, United Kingdom, Uzbekistan, Kosovo and released by the European Centre for Disease Prevention and Control (ECDC). The views and opinions of the author expressed herein do not necessarily state or reflect those of ECDC. The accuracy of the author's statistical analysis and the findings they report are not the responsibility of ECDC. ECDC is not responsible for conclusions or opinions drawn from the data provided. ECDC is not responsible for the correctness of the data and for data management, data merging and data collation after provision of the data. ECDC shall not be held liable for improper or incorrect use of the data.

Jesper Olsson, Gothenburg, July 2024

List of Acronyms & Abbreviations

Below is a list of the abbreviations for antibiotics and their antibiotic class.

	Name	*Class*
AMK	Amikacin	Aminoglycosides
AMP	Ampicillin	Penicillins
CAZ	Ceftazidime	Cephalosporins
CIP	Ciprofloxacin	Fluoroquinolones
CRO	Ceftriaxone	Cephalosporins
CTX	Cefotaxime	Cephalosporins
ETP	Ertapenem	Carbapenems
FEP	Cefepime	Cephalosporins
GEN	Gentamicin	Aminoglycosides
IPM	Imipenem	Carbapenems
LVX	Levofloxacin	Fluoroquinolones
MEM	Meropenem	Carbapenems
MFX	Moxifloxacin	Fluoroquinolones
NAL	Nalidixic acid	Fluoroquinolones
TOB	Tobramycin	Aminoglycosides

Next, a list of acronyms widely used throughout the text:

AMR	Antimicrobial Resistance
ANN	Artificial Neural Network
AST	Antimicrobial Susceptibility Testing
CPT	Class Pre-training
CV	Cross-validation
FFNN	Feed-forward Neural Network
MLM	Masked Language Modeling
MLP	Multilayer Perceptron
NCBI	The National Center for Biotechnology Information
NLP	Natural Language Processing
RPT	Random Pre-training
TESSy	The European Surveillance System

Contents

List of Figures

List of Tables

1

Introduction

Antimicrobial resistance (AMR) is a rising threat to global public health. In 2019, an estimated 1.27 million deaths were directly attributed to bacterial AMR, often called antibiotic resistance [1]. A 2016 report estimated that the global death toll could rise to 10 million by 2050 [2]. While the underlying cause of AMR is evolution by natural selection, the increase is driven by increased environmental exposure due to the misuse and overuse of antimicrobials in medicine and agriculture leading to increased selection pressure [3].

The increase in AMR makes it more difficult to treat infections. Until the process of determining the appropriate treatment is completed, *empirical* treatments are administered based on limited information [4]. Whether the empirical treatments are appropriate or not can significantly impact mortality [5]. This effect is observed both when empirical treatments are too broad, or not broad enough [6]. Additionally, the time from specimen collection to receiving appropriate treatment is an important factor in reducing mortality [7, 8].

The main process of determining an infective pathogen's resistance is called antimicrobial susceptibility testing (AST), and most commonly entails cultivating the bacteria in the presence of each antibiotic of interest. This is a time-consuming and costly process, meaning not all antibiotics will be tested. For resistant infections, multiple rounds of testing can therefore be required to find a suitable treatment alternative, further exacerbating the risks [9]. Therefore, models that can predict full resistance profiles from incomplete diagnostic information can play an important role in improving antibiotic resistance diagnostics, either by improving the empirical treatment or by speeding up the process of finding an appropriate treatment.

The basis of using statistical methods to model antimicrobial resistance is that the main spreading mechanism of antimicrobial resistance genes (ARGs) is via horizontal gene transfer, where ARGs are transmitted between individual bacteria [10]. The genes often spread as sets, which creates a non-uniform co-occurrence distribution of ARGs and, as a result, of observed resistance phenotypes [11]. Additionally, there can be dependencies on patient characteristics, e.g. age and gender, and more general characteristics, e.g. date and place of infection [12]. If statistical models can capture these co-dependencies, both within and between data modalities, accurate predictions of unknown phenotypes from limited information in a clinical setting could be possible.

Previous work on modeling antibiotic resistance for clinical decision support using machine learning has employed traditional methods such as Decision Trees, Random Forest, and simple neural networks [13]. However, these methods are rigid when it comes to handling missing data, making them ill-suited for making predictions based on input data with varying availability.

The *Transformer* is a deep learning architecture introduced in 2017 for language translation adept at handling varying input data [14]. It enabled efficient capture of context-dependent relationships between words in natural language processing (NLP). However, the mechanisms underlying the transformer have since been used for many other applications, such as chemical modeling and computer vision [15]. Beyond its ability to handle missing data, it is also efficient, flexible, and scalable. A transformer-inspired model has been successfully applied to predicting antibiotic resistance from patient data and incomplete AST results [16]. This thesis project aims to extend that work by incorporating genotype data into the process.

By using a transformer-inspired model in conjunction with data fusion - the practice of combining different types of data during some stage of its processing - a more comprehensive understanding of data dependencies is possible. Due to the complexity of the underlying processes, this becomes especially important when modeling biology [17]. In the case of AMR diagnostics, a multimodal transformer augmented by data fusion of available genotype, phenotype, and patient information could predict full resistance profiles. Here, genotype and phenotype are considered separate data modalities capturing resistance, while metadata such as patient information is assumed to be present and not considered a separate modality.

1.1 Aims

This thesis project aims to employ data fusion of genotype, phenotype and metadata in conjunction with a transformer-inspired deep learning model to improve AMR diagnostics in clinical settings, and it can be divided into three main subaims:

1. Develop, implement, and evaluate a multimodal transformer that uses data fusion to capture the dependencies between genotype, phenotype, and patient data to predict antibiotic resistance in *Escherichia coli*.

2. Investigate the training setup that best utilizes the available data.

3. Evaluate the observed benefit of multimodality in predicting antibiotic resistance.

1.2 Outline

This section will present the outline of this text. Chapter 2 presents the theoretical concepts required to follow the methodology, results, and conclusions of this work. A short description of antibiotic resistance is followed by a presentation of machine

learning, binary classification, and deep neural networks, including self-attention and the transformer encoder. Chapter 3 covers the methodology, describing the data and model and how it is trained and evaluated.

Chapter 4 first presents and discusses the results of experiments aimed to evaluate the model in clinically relevant scenarios and to determine the effect of pre-training strategies on fine-tuning performance. Then, results from experiments aimed at evaluating the effect of different levels of genotype data availability are presented before a final discussion of the results and their implications for future research. Finally, Chapter 5 sums up the implications of the results and draws conclusions on the aims and future research directions.

2

Theory

This chapter aims to present the theoretical concepts underlying this work and aid in understanding its contents.

2.1 Antibiotics and antibiotic resistance

The discovery and development of antibiotics in the 20th century led to a revolution in the medical sciences with wide-ranging impacts on society. Antibiotics are chemical substances that aim to combat infections either by killing (bactericidal) or inhibiting the growth of (bacteriostatic) bacteria. Antibiotics have been present in nature long before their "discovery" by humans, and most antibiotics used in clinics are derived from natural sources [18].

Antibiotics mainly differ from each other in what parts of the bacterial cell they target and are grouped by their *antibiotic class*. The antibiotics in this work are of five different classes: *Penicillins*, *Aminoglycosides*, *Cephalosporins*, *Carbapenems*, and *Fluoroquionolones*. Of these, Penicillins, Cephalosporins, and Carbapenems belong to a broader class called *β-lactams* that target the biosynthesis of peptidoglycan, a rigid component that makes up a vital part of the cell called the cell wall [19]. Aminoglycosides are broad-spectrum bactericidal agents that act mainly by inhibiting protein synthesis in the cell [20], while fluoroquinolones act by directly inhibiting DNA synthesis [21].

Like antibiotics themselves, resistance to them is not a new phenomenon [22]. Antibiotic resistance (AR) arises from the evolutionary selection pressure applied by antibiotics. With increased use of antibiotics comes an increased selection pressure, and, in turn, increased resistance. Since the widespread introduction of antibiotics in the second half of the 20th century, not only for medical use but also for agriculture, resistance has emerged rapidly and broadly [10].

The mechanisms of AR can be divided into three main types: first, minimizing the concentration of the antibiotic within the cell, either by preventing it from entering or by pumping it out; second, modifying the antibiotic target, often by genetic mutation; and third, by inactivating the antibiotic via chemical modification [23].

The sources of antibiotic resistance in a bacterium lie in its genome and occur in

two main ways. First, *point mutations* in the chromosomal DNA can give rise to modifications in cellular functions that help protect against antibiotics. These mutations are then transmitted to subsequent generations, which is where selection pressure comes in. Second, bacteria can take up whole resistance genes, for example, via small circular DNA molecules called *plasmids* [3].

Importantly, genetic material coding for resistance can be transferred between bacteria in a process called *horizontal gene transfer*, of which the spread of plasmids is the main mechanism [10]. If selection pressure is exerted on a bacterial population, bacteria carrying plasmids coding for resistance may fare better than their peers and spread them, leading to increased resistance in the population [24]. Commonly, plasmids carry multiple resistance genes, implying that some genes will co-occur more frequently than others [11].

2.2 Machine learning and neural networks

Machine learning (ML) refers to a broad range of techniques and algorithms that enable computer models to improve their performance on cognitive tasks by allowing them to iteratively learn from examples in problem-specific training data. This removes the need to pre-define rules for the model to follow when analyzing the data, increasing model flexibility and allowing for hidden patterns to be discovered, particularly in high-dimensional data [25]. Due to this flexibility, ML has seen wide application on a range of problems, such as product recommendation, diagnostics, chatbots, and fraud detection [26].

2.2.1 Deep neural networks

While there are many types of machine learning algorithms, those based on *deep neural networks* (DNNs) have gained particular popularity due to their flexibility and efficiency in analyzing large-scale data. These methods are commonly referred to as *deep learning* (DL) [27].

DNNs are based on artificial neural networks (ANNs), first developed in the mid-20th century to model how learning functions in the human brain as a network of interacting artificial neurons. DNNs are ANNs with multiple hidden layers of neurons. An example of this is the multilayer perceptron (MLP), also called a multilayer fully-connected feed-forward neural network (FFNN), visualized in Figure 2.1. Each neuron processes the outputs of neurons in the preceding layer and yields an output according to a biased weighted sum of the inputs passed through an activation function [28]. The activation function introduces non-linearity, allowing the model to capture more complex relationships. The number of neurons in the input and output layers is determined by the number of input variables, or *features*, and output variables. Commonly, there is one input neuron for each feature and one output neuron for each output variable.

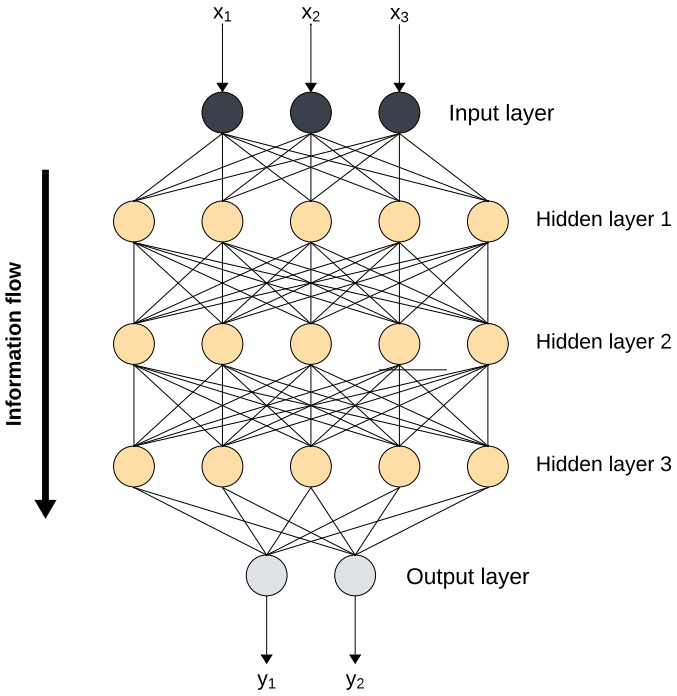


Figure 2.1: Multilayer perceptron with three hidden layers of five hidden neurons each. The network processes the inputs, x_i to yield the outputs, y_i.

Deep learning for classification

In multilabel classification with C labels, the output layer would be composed of C neurons. The outputs, $\mathbf{b} = \{b_1, \ldots, b_C\}^\top$, often called the *logits*, of the output layer are passed through a *softmax* layer that applies the function

$$\text{Softmax}(b_i) = \frac{e^{b_i}}{\sum_{c=1}^{C} e^{b_c}},$$

to each logit, yielding $\mathbf{p} = \{p_1, \ldots, p_C\}^\top$, where the value $p_c \in (0, 1)$ for label c can be interpreted as the predicted probability of label c. The predicted label is thus $\hat{y} = \underset{i}{\text{argmax}}\ \mathbf{p}$.

For binary classification, an output layer with one neuron can also be used. Then, the logit b is passed through the *sigmoid* function

$$\sigma(b) = \frac{1}{1 + e^{-b}},$$

yielding $p \in (0, 1)$ which can be interpreted as the predicted probability of the *positive label*, $y = 1$. The positive label is thus predicted if $p > 0.5$, equivalent to if $b > 0$.

2.2.2 Training neural networks

In the context of supervised learning, e.g. classification, neural networks are trained by backpropagating the output of a loss function of the error between predicted

and true labels and updating the parameters accordingly. Commonly, the loss is calculated as the *cross-entropy* (CE) between logits and the target label according to

$$L = \{l_1, \dots, l_N\}^\top, \quad l_n(\mathbf{p}_n, y_n) = -\log\left(p_{n,y_n}\right), \tag{2.1}$$

where $n = 1, \dots, N$ indexes across the N samples for which the loss is calculated, $\mathbf{p}_n$ contains the predicted probabilities for each label, and p_{n,y_n} is the predicted probability for the true label, y_n of sample n [29]. Commonly, the loss is then averaged across the samples. In the binary classification case, the binary cross-entropy (BCE) loss is calculated as

$$L(\mathbf{p}, \mathbf{y}) = \{l_1, \dots, l_N\}^\top, \quad l_n = -\left[y_n \log p_n + (1 - y_n) \log(1 - p_n)\right], \tag{2.2}$$

where $\mathbf{p}$ contains the predicted probabilities for the positive class for each sample and y_n is the true binary label, 0 or 1. BCE loss is revisited in Section 2.4.1 in the context of handling binary classification with imbalanced labels.

The model parameters are updated to minimize the loss using the gradient of the loss function. The scale of the parameter changes is regulated by a *learning rate*, γ. When a network has seen all samples in the dataset once during training, it is said to have completed one *epoch* of training. When training neural networks with large datasets, it is common to use *mini-batching*, a process where a training set with N examples is divided into $\lceil N/B \rceil$ batches with B examples, and parameters are updated after each batch by backpropagating the loss calculated from the batch examples. This manages memory usage and speeds up training [30].

2.3 Self-attention and the transformer

In 2017, Vaswani et al. published an influential paper proposing a deep learning architecture for sequence-to-sequence machine translation they called the *Transformer*. They assert that attention mechanisms can replace hitherto widespread convolution and recurrence mechanisms [14]. Since then, the model and models inspired by it have seen wide applications in areas such as natural language processing (NLP), computer vision, and the life sciences [15].

In particular, Bidirectional Encoder Representations from Transformers (BERT) [31] has been influential in adapting transformers to tasks other than sequence-to-sequence translation. BERT provides a framework for training a model based on the transformer encoder for a range of modeling tasks, such as classifying tokens or sequences, and serves as an inspiration for the transformer-based model developed in this work.

This section aims to explain the structure and mechanisms that enable a transformer model to effectively and efficiently capture information in sequence data. First, an overview of the architecture and how the components fit together is provided before the components are further explained separately.

2.3.1 General architecture

The transformer model, visualized in Figure 2.2 consists of three main parts: the tokenization, the embedding, and the encoding. The tokenization process consists of splitting up a sequence into parts, or *tokens*, and assigning each token an integer ID. In natural language, a tokenization process could, for example, involve splitting up a sentence into words and letting each word be a token. The token ID is determined by the token's location in the *vocabulary* and serves as an index to a lookup table of *token embeddings*. The vocabulary is commonly established to contain all the unique tokens in the data. Generally, an embedding is a learnable representation of a categorical value as vectors in a space of dimension d_e. A sequence of n tokens would yield n d_e-dimensional vectors, and since the embedding vectors of tokens exist in the same space, mathematical operations can be used to relate them. The sequence of token embeddings is passed through encoding, often consisting of multiple encoders stacked upon each other. It is within the encoders that the analysis of the sequence takes place.

2.3.2 Self-attention

Self-attention is the process of enriching token embeddings with information on how tokens in the sequence relate to each other. In the first step of self-attention, three linear transformations of the token embedding matrix $X \in \mathbb{R}^{n \times d_e}$ are carried out to yield the query, $Q = XW_Q \in \mathbb{R}^{n \times d_k}$; key, $K = XW_K \in \mathbb{R}^{n \times d_k}$; and value, $V = XW_V \in \mathbb{R}^{n \times d_v}$ matrices. The matrices $W_Q \in \mathbb{R}^{d_e \times d_k}, W_K \in \mathbb{R}^{d_e \times d_k}$, and $W_V \in \mathbb{R}^{d_e \times d_v}$ project the token embedding vectors into query, key and value space, respectively. Commonly, $d_k = d_v = d_e$. Then, *scaled dot-product attention* is applied by

$$\text{Attention}(Q, K, V) = \text{Softmax}\left(\frac{QK^\top}{\sqrt{d_k}}\right)V = AV,$$

where $A \in \mathbb{R}^{n \times n}$ is called the attention matrix. The attention matrix contains the *attention weights* that enrich the token embeddings with context from other tokens in the sequence. Decomposing the operations in the attention matrix, the operation $QK^\top \in \mathbb{R}^{n \times n}$ calculates the dot-product between each token's query vector and the key vectors of all tokens in the sequence to yield similarity scores that are then normalized by applying softmax along the rows. The scaling factor $1/\sqrt{d_k}$ is multiplied before the softmax operation to reduce the vanishing gradient problem of softmax when d_k is large [15].

Multi-headed self-attention

Commonly, transformer models use a variation of self-attention called *multi-headed self-attention*. The core attention mechanism remains the same as described above, but is conceptually conducted multiple times in parallel. This is done by splitting the embedding vectors of dimension d_e into h vectors of dimension $d_h = d_e/h$ that are then individually projected by the matrices $W_i^Q \in \mathbb{R}^{d_h \times d_k}, W_i^K \in \mathbb{R}^{d_h \times d_k}$, and $W_i^V \in \mathbb{R}^{d_h \times d_v}$ for head i and fed into h separate attention heads. After the attention step, the products of the attention heads are concatenated to yield vectors of

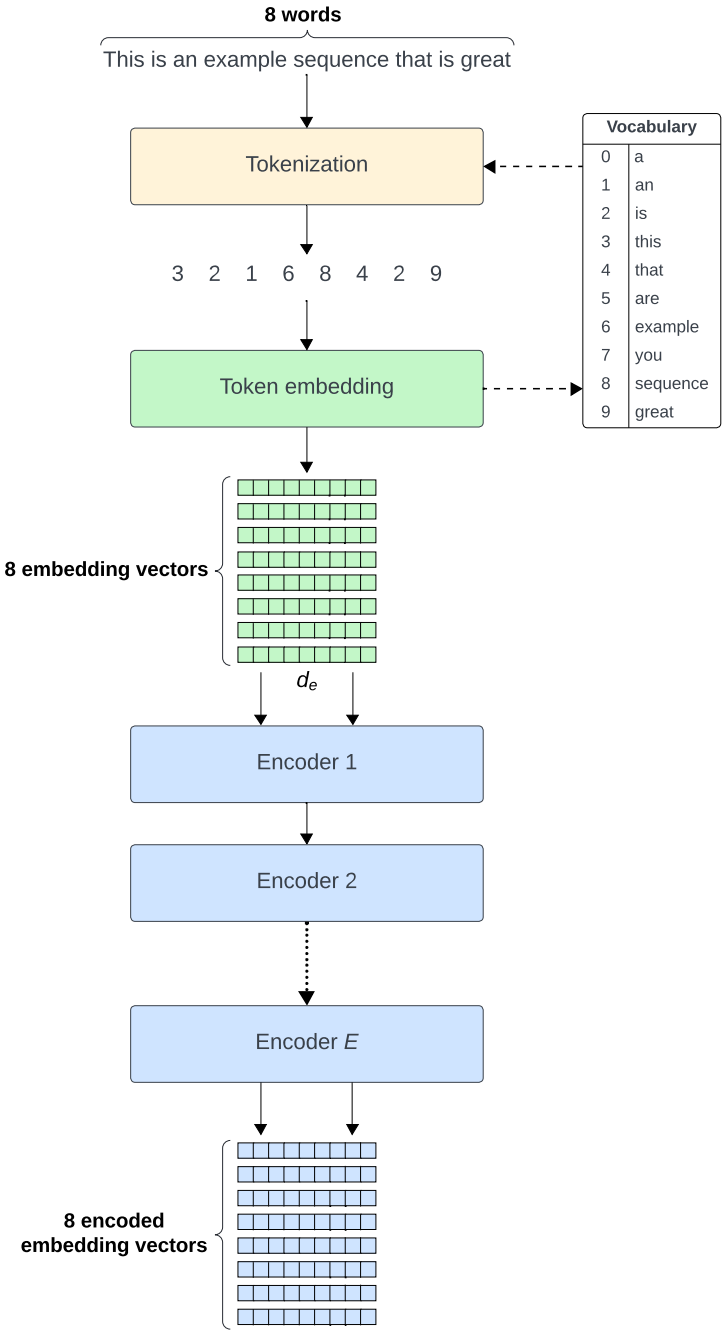


Figure 2.2: Overview of the transformer architecture. Words in the sentence are assigned integer IDs, or *tokens*, using a lookup table called a *vocabulary* before being projected to an embedding space to create vectors of dimension d_e. These embeddings are then fed through a stack of E encoders that process the sentence.

dimension $d_e = hd_h$ again. Multi-headed attention allows different heads to analyze the sequence in parallel with the aim of capturing different contextual aspects of the sequence [14].

2.3.3 Transformer encoder

The transformer encoder, visualized in Figure 2.3, first takes the input embeddings and splits them into equal-sized parts along the embedding dimension, corresponding to the number of attention heads. These parts are then individually projected into query, key, and value vectors that are passed through respective attention heads to enrich the embeddings with contextual information. The products of multi-headed attention are concatenated to form the original token embeddings. Next, residual connections are used to address the vanishing gradient problem of building deep neural networks. These work by adding the input embeddings to the enriched embeddings, and layer normalization is then applied before passing the embeddings through a position-wise feed-forward network (PFFN). The PFFN consists of two linear layers separated by a ReLU activation function and applies the transformation equally on each position in the sequence. Again, the residual connection and layer normalization step is applied to the output of the PFFN to yield the output embeddings of the encoder [15]. If multiple encoders are stacked, the output embeddings act as input embeddings in the following encoder.

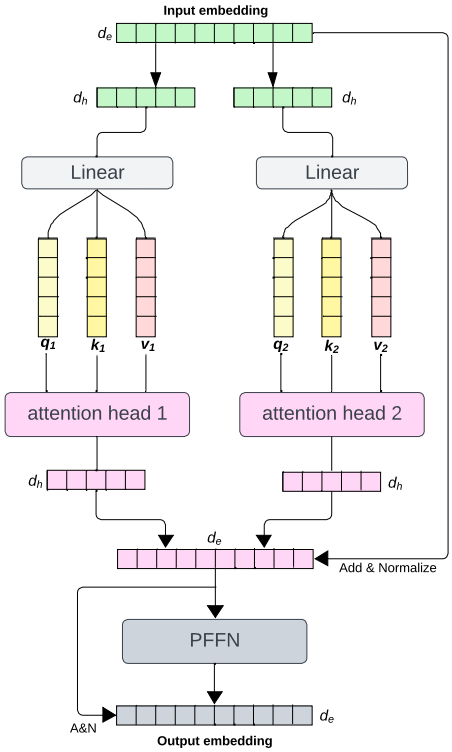


Figure 2.3: Overview of the transformer encoder. The input embedding gets split into parts that are then individually projected to query, key, and value vectors that are passed through self-attention. The attention-enriched embeddings are concatenated and passed through a position-wise feed-forward (PFFN) network that yields the output.

2.3.4 Pre-training and fine-tuning transformer models

A common practice in developing transformer models is to *pre-train* the model. Pre-training is usually done on tasks that the model is not directly intended to perform when deployed, but that improves generalizability and gives it relevant prerequisites for further training on downstream tasks, often called *fine-tuning* [32]. This is particularly common in NLP, and BERT was an early example showing how pre-training could enhance performance on downstream tasks [31]. Commonly, *self-supervised learning* is used for pre-training, since it makes use of large unlabeled datasets not directly tied to the downstream task. Self-supervised learning works by creating labels and supervised learning tasks inherent in the data. One such task is masked language modeling (MLM).

In modeling natural language, MLM entails presenting the model with a sequence, such as a sentence, and withholding (masking) select words of that sentence by replacing the tokens of the masked words with a special `MASK` token. The model is then asked to predict which tokens in its vocabulary are behind the masks. By treating this as a multi-class classification problem, and computing and backpropagating the loss, the aim is for the model to use self-attention to build a context-dependent understanding of how words relate to each other. Since this training process is highly scalable, the model can build this understanding from large volumes of text data from books or web sources. This understanding of the underlying language can then serve as a basis when the model is fine-tuned on other tasks, such as text classification, speeding up convergence and improving performance [32].

2.4 Binary classification on imbalanced data

In binary classification, there are only two labels, positive or negative. A prediction can therefore lead to one of four outcomes: a true positive (TP), where the positive label was correctly predicted; a true negative (TN), where the negative label was correctly predicted; a false positive (FP), where a positive label was predicted for a negative instance; and a false negative (FN) is where a negative label was predicted for a positive instance.

It is not uncommon for the distributions of the two labels to be imbalanced. An example is anomaly detection, where, per definition, there are many more negative examples than positives. Common evaluation methods of machine learning algorithms, such as cross-entropy, defined in Equation 2.2, and accuracy, weigh both labels equally, which may lead to difficulty in training the model and evaluating its performance. This section will describe alternative methods to counteract this.

2.4.1 Weighted binary cross-entropy loss

To account for the imbalance of labels in the training data, the loss calculated from samples can be weighted based on the proportion of their label. In the case of weighted binary cross-entropy, a weighting factor $\alpha \in [0, 1]$ is introduced for the

positive label, modifying the BCE loss defined in Equation 2.2 to yield

$$L(\mathbf{p}, \mathbf{y}) = \{l_1, \ldots, l_N\}^\top, \quad l_n = -\left[\alpha y_n \log p_n + (1-\alpha)(1-y_n)\log(1-p_n)\right], \quad (2.3)$$

where p_n is the predicted probability of the positive label and y_n is the true binary label. The aim of weighting the BCE loss is to push the model to pay more attention to examples of the minority label.

2.4.2 Evaluation metrics

There are many different metrics to evaluate model performance on a binary classification task, but they fall into three categories. Either they aim to capture overall model performance, performance on one label, or how the model balances inherent trade-offs between certain performance measures.

An example of the first category is also one of the most common metrics. The *accuracy* of a model is defined as

$$\text{Accuracy} = \frac{\text{TP} + \text{TN}}{\text{TP} + \text{TN} + \text{FP} + \text{FN}} \in [0, 1],$$

or, simply, as the proportion of predictions that were correct. While it is beautiful in its simplicity, accuracy can be misleading when labels are very imbalanced, since a model that always predicts the majority label will get a high accuracy score even though it would never capture an instance of the minority label. In cases of data imbalance, accuracy is best used in conjunction with other metrics, or not at all. In this thesis, accuracy will not be considered due to the large label imbalances visualized in Figures 3.1 and 3.2.

The metrics *sensitivity* and *specificity* measure the model's ability to capture the positive and negative instances, respectively. They are defined as

$$\text{Sensitivity} = \frac{TP}{TP + FN} \in [0, 1],$$
$$\text{Specificity} = \frac{TN}{TN + FP} \in [0, 1],$$

and can be seen as within-label accuracy. Another useful metric is *precision*, defined as

$$\text{Precision} = \frac{TP}{TP + FP} \in [0, 1],$$

which measures the accuracy of the model's positive predictions. These metrics are often presented together, or at least in pairs, since there are inherent trade-offs between them. For instance, a model that tends to overpredict the positive label would yield high sensitivity at the cost of low precision. An example of a metric that aims to capture this trade-off is the F_1-score, defined as the harmonic mean of sensitivity and recall, given by

$$F_1 = \frac{2}{\text{precision}^{-1} + \text{sensitivity}^{-1}} = 2\frac{\text{precision} \cdot \text{sensitivity}}{\text{precision} + \text{sensitivity}} \in [0, 1].$$

3

Methods

To achieve the aims stated in Section 1.1, a fusion transformer was implemented in Python v3.11.5 using PyTorch v2.1.0. This chapter describes the methodology of the thesis project, first by describing the data and then the model and how it was trained and evaluated. The code for the implementation, training, and evaluation of the model is available in the following GitHub repository: https://github.com/jespeols/M-TARP

3.1 Datasets and data sources

In this work, two datasets are constructed by combining samples from two data sources, here referred to as TESSy and NCBI. Since these data sources differ in composition and pre-processing, they will be outlined in separate subsections. In both cases, data is limited to the pathogen *Escherichia coli* (*E. coli*), and the final product of pre-processing is a dataframe with rows corresponding to individual samples or patients, with separate columns for the metadata attributes and the unimodal lists containing the available genotypes or phenotypes in the form of names of resistance genes and point mutations and antibiotic susceptibility testing (AST) results, respectively. The date and geographical information were reduced to year and country.

3.1.1 TESSy

The European Surveillance System (TESSy) collects data from clinical settings in European countries as part of a disease surveillance effort [33]. The raw data lists over 24 million AST results for multiple pathogens. For *E. coli*, there are ~14 million AST results across 26 unique antibiotics, primarily from bloodstream infections. The TESSy data contains no genotype data, making it unimodal phenotype with each row corresponding to an antibiotic tested on a bacterial isolate from a patient. The row contains the antibiotic tested and its result, with associated metadata and some identifiers (IDs). This means that the data must be aggregated to achieve a dataset listing all AST results for each isolate.

Pre-processing

By combining the date and country of the test with three IDs, namely the Laboratory ID (unique to each lab in the country), the Patient ID (unique to each patient within the lab), and the Isolate ID (unique to each isolate within the lab and year), we obtain a unique sample identifier associated with a unique patient and isolate with which we can aggregate AST results across antibiotics. AST results were limited to the binary susceptible (S) or resistant (R) designation. When doing the aggregation of test results, they are stored as a list of strings of the format "ABC_X", with "ABC" corresponding to the three-letter abbreviation of the antibiotic provided in the dataset, and "X" corresponding to the S/R designation.

Before doing the aggregation, samples were dropped if they had a missing isolate ID or missing gender or age data. Additionally, duplicate tests with the same ID and AST result were dropped. After the pre-processing and aggregation, $N_{u,p} = 1,440,857$ samples remained. Hereafter, these aggregated samples will be referred to as "isolates".

Antibiotic distributions

The TESSy dataset contains AST results from 26 unique antibiotics. However, since the specific antibiotics tested vary greatly between isolates, there are imbalances in the data distributions. Therefore, several antibiotics in the data were excluded. The selection process is described in Appendix A. In the end, eleven of the 26 antibiotics were excluded, leaving the fifteen antibiotics whose names, abbreviations, and class affiliations are presented in Table 3.1. The classes aminoglycosides and carbapenems are each represented by three antibiotics, while fluoroquinolones and cephalosporins are each represented by four and penicillins by one.

Table 3.1: Abbreviations and class affiliations of the fifteen selected antibiotics.

Antibiotic	Abbreviation	Antibiotic class
Amikacin	AMK	Aminoglycosides
Ampicillin	AMP	Penicillins
Ceftazidime	CAZ	Cephalosporins
Ciprofloxacin	CIP	Fluoroquinolones
Ceftriaxone	CRO	Cephalosporins
Cefotaxime	CTX	Cephalosporins
Ertapenem	ETP	Carbapenems
Cefepime	FEP	Cephalosporins
Gentamicin	GEN	Aminoglycosides
Imipenem	IPM	Carbapenems
Levofloxacin	LVX	Fluoroquinolones
Meropenem	MEM	Carbapenems
Moxifloxacin	MFX	Fluoroquinolones
Nalidixic acid	NAL	Fluoroquinolones
Tobramycin	TOB	Aminoglycosides

In Figure 3.1, the prevalences of the antibiotics and the label imbalances are visualized. The prevalence distribution is highly uneven, with only six of the fifteen antibiotics present in more than a majority of isolates and, except for AMP, the labels of all antibiotics are skewed towards susceptible, with some antibiotics, such as MEM and ETP, present as susceptible at a rate ~100%.

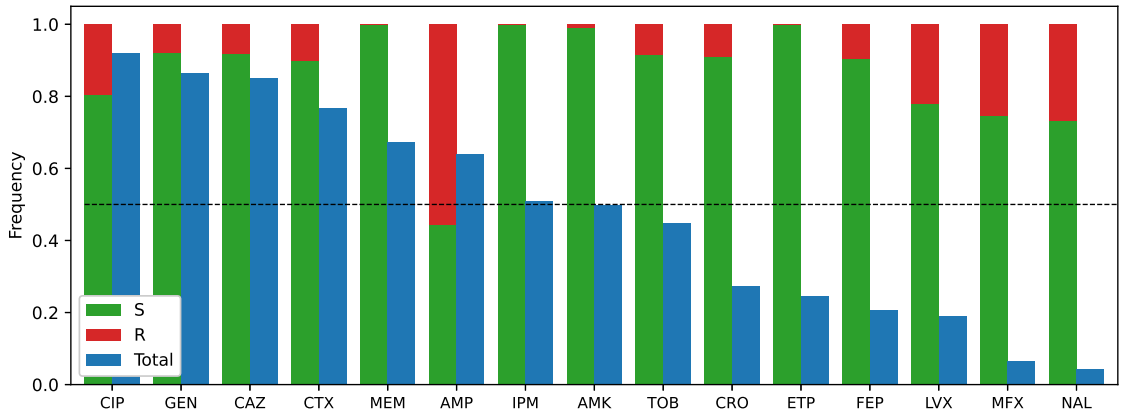


Figure 3.1: Frequency of antibiotics in blue, as the share of total number of isolates, and S/R label imbalance for each antibiotic in the pre-processed TESSy dataset. The prevalence varies greatly between antibiotics. Most antibiotics are majority-susceptible, with some near 100% susceptibility rates.

3.1.2 NCBI

The National Center for Biotechnology Information (NCBI) Pathogen Detection Project provides data from pathogenic isolates from clinical and environmental sources worldwide. By applying the tool AMRFinderPlus on genomic sequences in the Pathogen Detection pipeline, NCBI provides a standardized set of identified resistance genes. Each row of the raw dataset contains a list of the annotated AMR genotypes as well as the date and location of the sample [34]. The database is updated on a daily basis, and the version used in this thesis project is 4460, uploaded to NCBI on April 25th, 2024 [35]. For *E. coli*, the NCBI dataset contains 375,000 isolates. Notably, 7,726 of these samples also contain phenotype information in the form of AST results for a varying set of antibiotics. This subset of the NCBI dataset can be called *multimodal*, since it contains both genotype and phenotype information, while the remaining subset can be called *unimodal* genotype.

Pre-processing

Since the raw NCBI dataset organizes the available genotype, phenotype, and metadata by isolates, the pre-processing mainly consists of filtering. Isolates that contain no genotype information or that were collected before the year 1970 were dropped. Many isolates have at least one metadata attribute missing, but some have both missing. Due to the smaller dataset size, these isolates were not dropped, as in the TESSy case, but if isolates contain only one genotype and no other information, they were dropped. The number of unique antibiotics and AST result designations

in the NCBI dataset are larger than in TESSy. The phenotypes were reduced to those with the antibiotics included in TESSy, and the designations S or R. They were then converted to the format "ABC_X" used for TESSy.

The NCBI dataset provides two sets of genotypes for each isolate, *AMR genotypes* and *AMR genotypes core*. The *core* set is a curated list that contains AMR-specific genotypes. The other set also contains "plus" genes that either affect other phenotypes, such as virulence, or affect AMR but where their presence/absence is unlikely to switch the AMR phenotype. Therefore, the *core* set of genotypes was chosen [36].

The dataset contains 1,476 unique genotypes, both point mutations and genes. They occur not as DNA sequences, but as annotated names, such as the gene *sul1* or the point mutation *glpT_E448K*. In the data, genes most commonly occur as *complete* assembly alignments, where the BLAST alignment covers >90% of the corresponding reference protein. However, genotypes can also occur with other assembly designations, such as *partial* or *partial end of contig*, where the alignment covers <90% of the reference protein [36]. In this work, only *partial end of contig* alignments are included beyond *complete* alignments, since these misalignments are more likely to be caused by a sequencing or assembly issue than for *partial* alignments. Once the other alignment variants were filtered out, the genotypes were checked for redundancy to ensure that unique genes didn't occur both as *complete* and as *partial end of contig*. If they did, only the *complete* alignment was kept.

After pre-processing, 365,951 isolates remained, of which $N_m = 6,494$ were multimodal, leaving $N_{u,g} = 359,457$ unimodal genotype samples.

Antibiotic and geographic distributions

In pre-processing, the phenotypes in the NCBI dataset were reduced to those with antibiotics also present in the TESSy dataset. However, three antibiotics (AMC, OFX, and TZP) are absent, leaving 23 unique antibiotics in the NCBI dataset. As in the TESSy dataset, the prevalence and class balance of the antibiotics are skewed, and the exclusion of antibiotics, as described in Appendix A, focuses on the distributions in the NCBI dataset. The distributions of the fifteen antibiotics are visualized in Figure 3.2. The prevalence distribution is staggered, with five antibiotics present at a rate >90%, eight antibiotics at ~50%, and the remaining ten at rates <15%. The classes are, in general, less imbalanced than in the TESSy dataset, with five antibiotics having majority-resistant skews. However, multiple antibiotics are present as resistant at a rate >95%.

While the TESSy dataset contains data solely from European countries, the data in NCBI is worldwide. However, the 155 unique countries are far from equally distributed. To investigate this, the country label for each isolate was mapped to one of the following regions: North America, South America, Europe, Africa, Asia, or Oceania, and the frequency distribution was computed both for the complete dataset and the multimodal subset. These distributions are presented in Figure 3.3, and show that North America is the most common region in the overall dataset, accounting for 40% of the samples, and by far the most dominant region in the

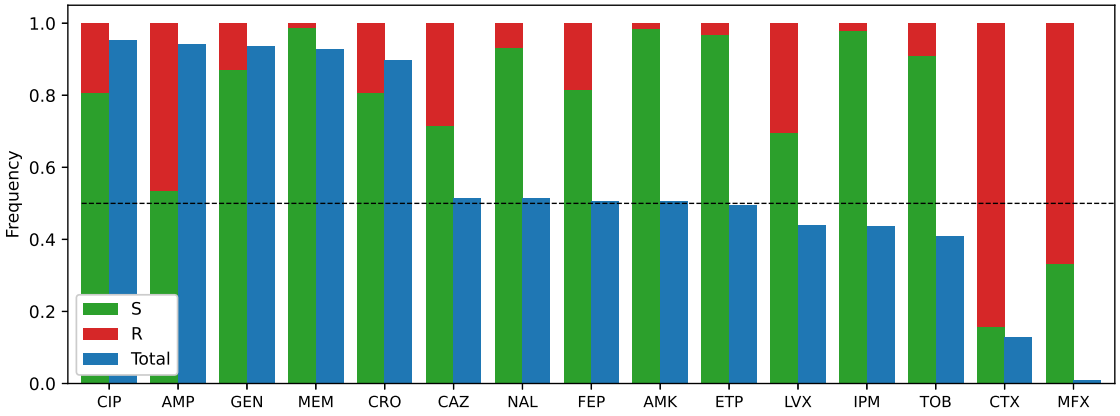


Figure 3.2: Frequency of antibiotics in blue, as the share of total number of multimodal isolates, and S/R class imbalance for each antibiotic in the pre-processed NCBI dataset. There are significant differences in prevalence and while the spread in class-imbalance is greater than in TESSy, many antibiotics are skewed toward susceptible.

multimodal dataset, where North America accounts for almost 90% of isolates and Europe accounts for ~10%. Indeed, the United States accounts for 44 and 86 percent of the overall and multimodal isolates, respectively. Notably, Oceania, Africa, and South America are hardly present in the multimodal subset. In the multimodal subset, the rate of isolates with a missing country label is much lower than in the overall dataset.

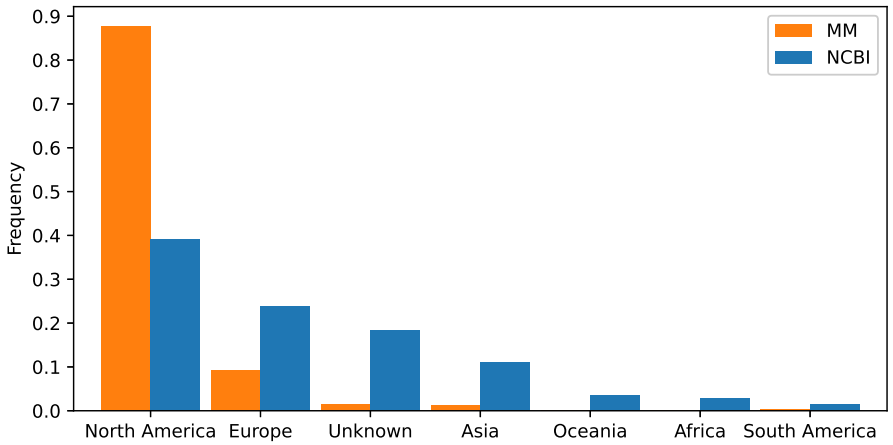


Figure 3.3: Frequency of regions for isolates of the overall NCBI dataset and multimodal subset. North America is the most prevalent region, accounting for almost 90% of the multimodal samples.

3.1.3 Constructing the datasets

From the two data sources NCBI and TESSy, two datasets were constructed: one containing unimodal samples and the other containing multimodal samples. The multimodal dataset is the subset of NCBI containing isolates with both genotype

and phenotype data. Thus, the multimodal dataset will consist of $N_m = 6,494$ samples. Table 3.2 presents an example isolate from the multimodal dataset.

Table 3.2: An example isolate from the multimodal dataset.

Year	Country	Age	Gender	Genotypes	Phenotypes
2013	Germany	-	-	blaTEM-1, tet(A)	CIP_S, LVX_S, NAL_R, NOR_S

The unimodal dataset consists of both unimodal genotype *and* unimodal phenotype samples, and will be called the *stacked* unimodal dataset. Table 3.3 presents three example isolates from this dataset. The isolates contain metadata and either genotype or phenotype data. Some genotype samples lack year or country labels, while none of them contain age and gender information. The stacked uni-

Table 3.3: Three example isolates from the stacked unimodal dataset. Dashes indicate missing labels.

Year	Country	Age	Gender	Genotypes	Phenotypes
2016	UK	-	-	pmrB_Y358N, glpT_E448K	-
-	Japan	-	-	tet(A), blaTEM-1	-
2015	UK	55	F	-	AMP_R, CIP_S, CRO_S, GEN_S

modal dataset is created by pre-processing the genotype and phenotype isolates separately before stacking them. The number of samples in the stacked unimodal dataset is given by $N_u = N_{u,g} + N_{u,p} = 359,457 + 1,440,857 = 1,806,808$. There is an unbalance in the composition, with frequencies $f_p = N_{u,p}/N_u = 0.797$ and $f_g = N_{u,g}/N_u = 1 - f_p = 0.203$.

Preparing the datasets for training and evaluation

When the data is prepared for training and evaluation, sequences are first constructed by collecting the available data in a list. The sequences are then tokenized using a vocabulary that maps each unique data value to an integer ID called a token. The tokenization process is illustrated in Figure 2.2. After the sequence is tokenized, a special `CLS` token that will be trained to act as an aggregated representation of the sequence is prepended to the tokenized sequence. Finally, the sequence, with L tokens, is padded to a maximum sequence length, $L_{\max}$, with $L_{\max} - L$ `PAD` tokens that do not contribute to self-attention. In cases of missing metadata, there is no specific token to indicate this. Instead, `PAD` tokens are used and the absence is implied.

3.1.4 Masking strategies

The classification tasks in this work are inspired by masked language modeling (MLM), described in Section 2.3.4. The selection of tokens to be masked can be

done in different ways but relies on two approaches: random or systematic masking.

Random masking is performed defining a probability, p, for each token to be masked. Systematic masking takes into account inter-token relationships when selecting tokens to be masked. An example in natural language would be to select adjacent words, since they are usually related grammatically. Here, the systematic selection is based on antibiotic class. For each sequence, k_c classes are randomly chosen to be kept, and any antibiotic not of those classes are masked. While the number of antibiotics present will vary between classes for a given sequence, all classes represented by at least one antibiotic are equally likely to be selected. An important aspect of a masking strategy is the difficulty of the resulting classification task. For the random masking strategy, the difficulty can be tuned by changing p, while the class masking difficulty is tuned by changing k_c.

In this work, masked genotype and phenotype tokens are replaced with separate mask tokens, `GENE_MASK` and `AB_MASK`. Masking is performed during the dataset preparation step, described in Section 3.1.3, on genotype and phenotype subsequences separately before the final sequence is constructed. In this work, *dynamic masking* is used, where the training set is re-masked at the start of each epoch. The validation set is only masked once.

3.2 Model

The model architecture is visualized in Figure 3.4 along with the training workflow described in Section 3.3. The core model component is the stacked set of E transformer encoders, described in Section 2.3, that take the embedded sequences as input and apply multi-headed attention. Added to the standard token embedding is a *token type embedding* used to indicate whether a token is metadata (0), genotype (1), or phenotype (2). The attention-enriched embeddings are then fed as input to neural networks that perform the downstream classification task. In this case, the model can perform two similar, but different, masked language modeling (MLM) tasks.

The first task is the multi-class classification of genes, which is performed by a single feed-forward output layer that maps the encoded embeddings, with dimension d_e, to the vocabulary, with size n_v, creating the classification logits. The second task is the binary antibiotic resistance classification of susceptible (S) or resistant (R). This is performed by a neural network specific to each antibiotic. These networks take the attention-enriched embedding of the `CLS` token as input, transforms it to a space with hidden dimension d_H, here chosen as $d_H = d_e$, and apply ReLU activation and layer normalization before a final output layer produces the classification logit. The `CLS` token is trained to capture sequence-wide information via self-attention that will be used to classify individual antibiotics. This is called *CLS pooling* and was introduced in BERT [31].

Table 3.4 contains the chosen model parameters. The vocabulary size n_v is given by the number of unique tokens in the data, and the maximum sequence length, $L_{\max}$,

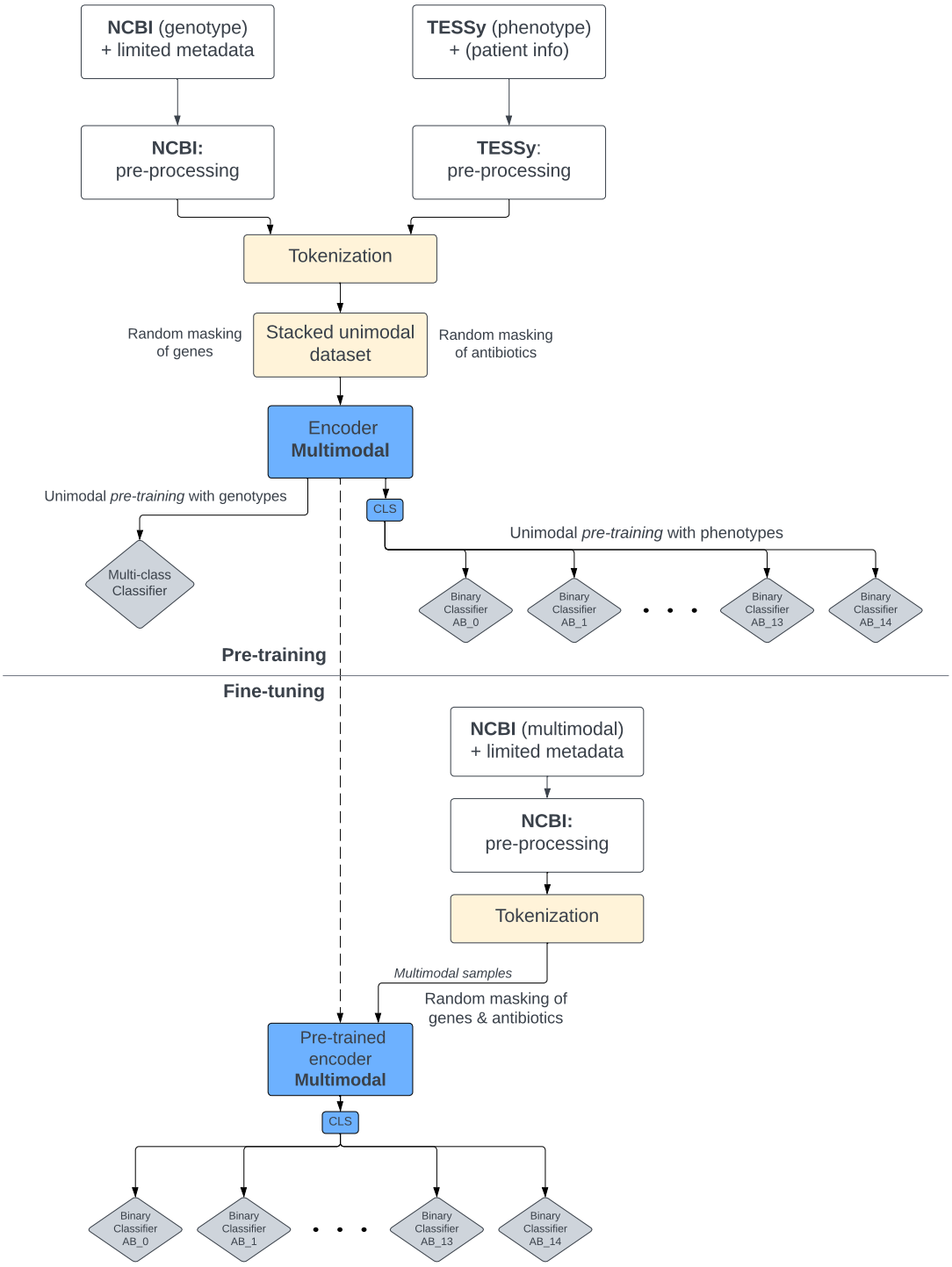


Figure 3.4: Overview of the model architecture, data preparation, and training workflow for the multimodal fusion transformer developed in this work. The model consists of a set of stacked transformer encoders feeding into two sets of classifiers for two separate MLM tasks. The first is a multi-class output layer that predicts masked genes and the other is a set of neural networks specific to each antibiotic classifying phenotypes. The pre-training uses a stacked unimodal dataset with samples from NCBI and TESSy, while the fine-tuning task uses the multimodal subset of NCBI. In pre-training, both MLM tasks are trained in parallel, while only the phenotype classification task is trained during fine-tuning.

is simply the longest in the data after the sequences are prepared.

Table 3.4: Model parameters used in this work. $L_{\max}$, n_v, and n_{ab} are determined by the data, while the rest are hyperparameters.

Parameter	Notation	Value
Maximum sequence length	$L_{\max}$	56
Vocabulary size	n_v	1586
Embedding dimension	d_e	512
Feed-forward dimension	d_f	512
Number of heads	h	4
Number of encoder layers	E	6
Dropout probability	p_d	0.1
Hidden dimension (ab classifier)	d_H	512
Number of antibiotics	n_{ab}	15

3.3 Training the model

The model described in Section 3.2 was implemented using Python v3.11.5 and Pytorch v2.1.0 [37] and trained on an NVIDIA Tesla V100 GPU using the `AdamW` optimizer with weight decay, $\lambda = 0.01$. To ensure reproducibility, random states were specified for the PyTorch and NumPy packages according to the respective documentation.

The approach to training the model is inspired by the pre-training and fine-tuning approach described in Section 2.3.4. While the pre-training and fine-tuning tasks are highly similar, the approach is relevant due to the large volumes of unimodal data (genotypes *or* phenotypes), compared to the small volumes of multimodal data (genotypes *and* phenotypes). The multimodality of the data defines the downstream task of predicting phenotypes from multimodal data. The pre-training aims to make use of these large volumes of unimodal data.

Therefore, the training of the model is divided into two stages. First, the model is pre-trained in the sense that it trains on unimodal data in the stacked unimodal dataset. During this step, the model trains on the tasks of multi-class gene classification and the multiple binary classification of antibiotic resistance in parallel. Then, the model is fine-tuned for the downstream task of multimodal antibiotic resistance classification by further training on the multimodal data. The model is not trained on the gene classification task in the fine-tuning step.

3.3.1 Pre-training

During pre-training, the stacked unimodal dataset was randomly divided into a train-val split with 90% used for training. Before this, the samples in the dataset were shuffled to get batches with both unimodal genotype and phenotype samples.

Pre-training was performed with batch size, $B = 32$, and a learning rate, $\gamma = 2 \cdot 10^{-5}$ for a maximum of 100 epochs or until the early stopping patience of 5 was surpassed. Early stopping is triggered when the loss calculated from the validation data has not improved for a fixed number of epochs, the so-called patience. This is to prevent overfitting, where the model becomes overly biased toward the training data and less adept at generalizing. If early stopping was triggered, the model parameters and results from the epoch with the lowest validation loss were saved.

For the genotype prediction task, cross-entropy (CE) loss, as defined in Equation 2.1 was used. Weighted binary cross-entropy (BCE) loss, as defined in Equation 2.3, was used for the phenotype prediction task, and individual loss functions were defined for each antibiotic to allow for label weights α_i specific to each antibiotic. However, during the pre-training, no label weighting was applied, and $\alpha_i = 0.5$ for all $i = 1, \ldots, n_{\text{ab}}$.

To fulfill the aim of finding the optimal training strategy for the model, six different pre-training variations were performed with two masking strategies at three difficulties. These are presented in Table 3.5.

Table 3.5: Parameters used for the different masking strategies during pre-training. p_{g} is the probability for each genotype to be masked, while p_{ab} is the masking probability for phenotypes during pre-training with random masking (RPT) and k_{c} is the number of known classes during pre-training with class masking (CPT).

Name	p_{g}	Phenotype
Easy RPT	0.25	$p_{\text{ab}} = 0.2$
Medium RPT	0.5	$p_{\text{ab}} = 0.5$
Hard RPT	0.75	$p_{\text{ab}} = 0.8$
Easy CPT	0.25	$k_{\text{c}} = 3$
Medium CPT	0.5	$k_{\text{c}} = 2$
Hard CPT	0.75	$k_{\text{c}} = 1$

3.3.2 Fine-tuning

During fine-tuning, the data was trained and evaluated using five-fold cross-validation (CV). This means that the multimodal dataset was first divided into five parts, the model then trained on four of these parts and evaluated on the remaining part. The fine-tuning process was then re-done for five runs with alternating, non-overlapping, evaluation parts, yielding five independent samples of the model performance. From this data, average model performance across folds can be computed along with the standard deviation. Comparing these measures between models allows for better-founded conclusions than comparing measures from one random split of the data.

Before the data was divided into folds, the samples in the dataset were shuffled.

Fine-tuning was performed with $B = 16$ and $\gamma = 1 \cdot 10^{-5}$ for a maximum of 100 epochs or until the early stopping patience of 10 was surpassed. Weighted BCE loss, as defined in Equation 2.3, was used with individual weights presented in Table B.1.

3.4 Evaluating the model

This section describes the methodology of the experiments performed to evaluate the model per the aims described in Section 1.1. As was described in Section 3.3.2, the model was evaluated by fine-tuning the model using a five-fold CV of the multimodal dataset.

3.4.1 Effect of pre-training strategies

To investigate the best strategy for pre-training the model, models pre-trained according to the strategies in Table 3.5 will be fine-tuned and evaluated along with a model without pre-training. This approach aims to isolate the effect of pre-training on the model's performance on downstream tasks. The evaluation consists of two fine-tuning experiments evaluating the model's performance on datasets prepared with random masking and class masking. The first task aims to evaluate the model's ability to act as a drop-in replacement for individual AST results across a wide range of combinations of available diagnostic data, while the second task aims to evaluate the model's ability to predict antibiotic resistance across antibiotic classes. In both experiments, genotypes were randomly masked with $p_{\text{g}} = 0.5$. This withholding of genotype information fulfills two purposes: first, to enable the learned ability of genotype inference through pre-training to be influential; and second, to simulate realistic scenarios where all genotype information may not be available.

Additionally, to investigate how model performance is impacted by the size of training data, and if this impact varies depending on the pre-training strategy, the pre-trained models and a model without pre-training were fine-tuned separately on training sets reduced to 1%, 5%, 10%, 20%, and 30% of the full dataset (6,494 isolates). Datasets reduced to these fraction levels will be referred to by their share. The fine-tuning was conducted with a five-fold CV and the reduction was performed after the data was folded. Since the samples in the dataset are randomly shuffled before divided into folds, the isolates included in the reduced training set could be selected non-randomly to ensure that, when the model is fine-tuned on a larger set, e.g. 5% of the dataset, the 5% share will include all isolates in the 1% share. This approach aims to simulate the process of gathering more data for training. The fine-tuning was conducted with both random and class masking for phenotypes and $p_{\text{g}} = 0.5$.

3.4.2 Effect of genotype data availability

To investigate the effect of genotype data availability on performance, the best-performing CPT and RPT models on the class inference experiment described in the section above, as well as a model without pre-training, were fine-tuned on that

same task ($k_c = 1$), at varying levels of available genotype information. The models were selected based on F_1 score. Additionally, *unimodal phenotype* models were included to isolate the role that genotype pre-training plays in fine-tuning.

The unimodal phenotype models have the same architecture and model parameters as the multimodal models but are only pre-trained on the phenotype task using the TESSy dataset. This means that it is only capable of handling the metadata and the phenotypes during fine-tuning. Unimodal models were pre-trained with the same set of pre-training strategies, i.e. the phenotype column in Table 3.5, fine-tuned on the $k_c = 1$ class inference task with five-fold CV along with a model without pre-training. Then, the best CPT and RPT models were selected for comparison with the multimodal models.

For the evaluation, the multimodal models were fine-tuned using five-fold CV at varying levels of available genotype data regulated by the masking probability, p_g, at 0%, 25%, 50%, 75%, and 100%, as well as in a scenario where the genotype data was not included at all. In that case, the dataset was pre-processed only with metadata and phenotype information, simulating the multimodal model being fed unimodal data and enabling a direct comparison with the unimodal model.

4

Results

This project aimed to develop, implement, and evaluate a fusion transformer to predict antibiotic resistance from genotype, phenotype, and metadata. In chapter 3, the model, training strategy, and data were presented, and this chapter aims to present the findings from evaluating the model. First, results from experiments that compare the effect of pre-training strategies will be presented, including what happens with reduced training data. Then, results from an experiment investigating the value of multimodality and the effect of genotype data availability are presented. Finally, an overarching discussion of the results and their implications for future research will round out the chapter. Whenever performance is broken down and presented by antibiotic, only a selection of metrics and antibiotics will be included to improve clarity and interpretability by focusing on the most informative results. Results for all antibiotics are provided for each experiment in Appendix C.

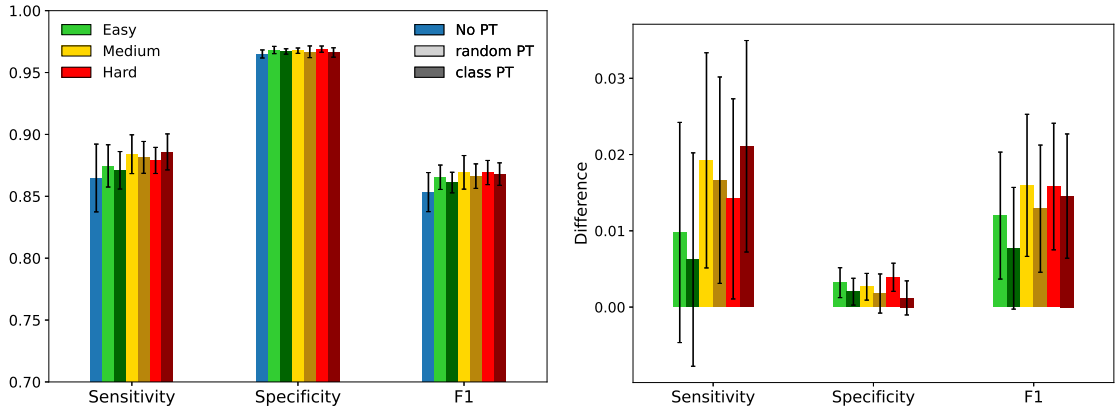
4.1 Pre-training strategies

To investigate the best strategy for training the model, models pre-trained according to the strategies in Table 3.5 were fine-tuned and evaluated along with a model without pre-training. Two fine-tuning evaluations were performed: one with random phenotype masking and one with class masking.

4.1.1 Random masking

Figure 4.1a shows performance from fine-tuning using random phenotype masking with probability $p_{\text{ab}} = 0.75$ as average sensitivity, specificity, and F_1 score across the five folds, with standard deviations as error bars. Performance is high across all metrics and models, with specificity above 95% and sensitivity above 85% for all models. Performance differences between models are small and while the variations in performance across folds are small, they are large in comparison to the differences in average performance between models.

Figure 4.1b shows the difference in average performance relative to the model without pre-training, with the error bars showing the standard error (SE) of the differ-



(a) Absolute value results **(b)** Results relative to `No PT`

Figure 4.1: Results from fine-tuning pre-trained models and a model without pre-training on the mulitmodal dataset with $p_{\text{ab}} = 0.75$ and $p_{\text{g}} = 0.5$. The green, yellow, and red colors indicate pre-training difficulty. Bars of models pre-trained with random masking have bright colors while bars of models pre-trained with class masking have dark colors. Panel a) shows average performance across the folds while panel b) shows the performance of pre-trained models relative to `No PT`. The bars of the randomly initialized model without pre-training are blue. Error bars indicate standard deviation in panel a) and standard error in panel b).

ences between the averages, given by

$$\text{SE} = \sqrt{\frac{\sigma_{m_1}^2 + \sigma_{m_2}^2}{n}} \tag{4.1}$$

where $\sigma_{m_1}^2, \sigma_{m_2}^2$ are the metric variances for the models compared and $n = 5$ is the sample size, i.e. the number of folds. There is a clear trend of improved average performance for pre-trained models, with a tendency for more difficult pre-training to outperform easier pre-training. Focusing on the sensitivity, `Hard CPT` performs the best, but the model striking the best balance between sensitivity and specificity appears to be `Medium RPT`.

Due to the relatively high variation in performance for the models, and for `No PT` in particular, the SEs are high in relation to the differences in average performance. However, in the case of F_1 scores, the SEs are lower, increasing the confidence that the performance gains of pre-training are significant.

Importantly, the results in Figure 4.1 are calculated across all antibiotics, and further insight into differences in model performance between pre-training strategies may be found by breaking down the performance by antibiotic. This analysis is presented for sensitivity and the antibiotic classes carbapenems, aminoglycosides, and penicillins in Figure 4.2. Sensitivity for all antibiotics are available in Figure C.1. The antibiotics are sorted from left to right by descending share of susceptible isolates, meaning the most imbalanced antibiotics are to the left. For reference, Figure 3.2 visualizes the label imbalances for all antibiotics in the fine-tuning dataset. The antibiotics whose resistance is hardest to capture are among those that are most

imbalanced. The performance variation is also higher for these antibiotics. While performance differences between pre-trained models and `No PT` are small, a trend of higher performance for pre-trained models on the more imbalanced antibiotics, and on GEN, can be observed.

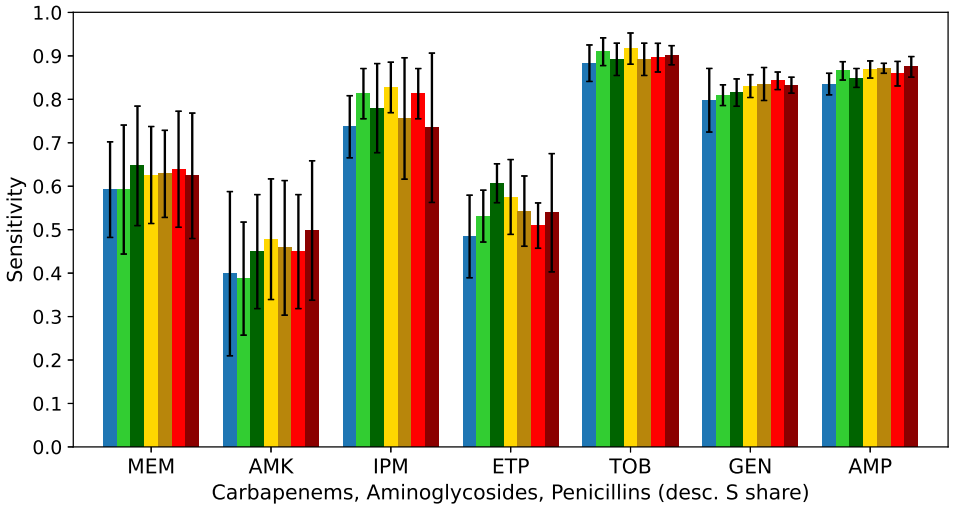


Figure 4.2: Results from fine-tuning with $p_{\text{ab}} = 0.75$ broken down by antibiotic and visualized for a selection of classes. Antibiotics are sorted from left to right by decreasing f_{S}, or share of susceptible isolates, meaning the most imbalanced antibiotics are to the left.

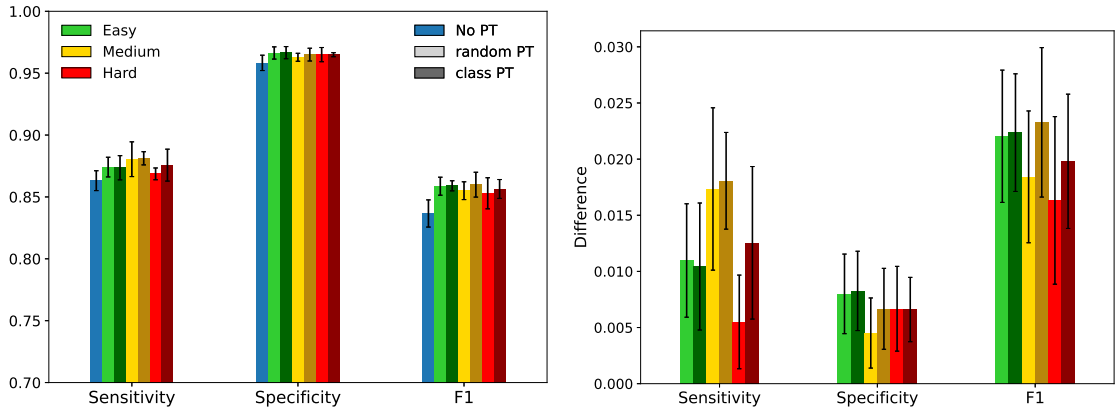
The results in Figure 4.1 demonstrate that the model can use available metadata and limited genotype and phenotype data to accurately predict antibiotic resistance. It is capable, both at capturing resistance even when it is rare, and in achieving the high specificity scores expected from the label imbalances. Importantly, though, performance varies between antibiotics, mainly by lower sensitivity on the most imbalanced antibiotics. The gaps between pre-trained models and `No PT` in Figure 4.2 tend to be larger for the more imbalanced antibiotics, suggesting that pre-training is especially beneficial for capturing resistance when it is rare.

There are large variations in model performance between folds, but Figure 4.1 shows that this variation is higher for `No PT`. Part of these variations could be related to the expected variation in difficulty between the validation sets due to the small number of samples and large diversity in data combinations. If so, the performance variation between folds would be similar for the models, since they would all be expected to perform worse on more difficult data. Further analysis that takes the difficulty of samples into account could be conducted to investigate if this is the case.

4.1.2 Class masking

Figure 4.3 shows results from fine-tuning using class masking with $k_{\text{c}} = 1$, meaning that the only phenotypes not masked are of the antibiotics present of one randomly selected antibiotic class. Notably, performance on the cross-class prediction task is

highly comparable to the random masking task, though the gap between pre-trained models and `No PT` is higher for specificity and, in turn, F_1. The best-performing models on sensitivity are those pre-trained at medium difficulty, with `Medium CPT` yielding higher specificity than `Medium RPT`. While the models pre-trained on easy difficulty are on par with `Medium CPT` on F_1, they are worse at capturing resistance. Given that the task of capturing resistance is more difficult, `Medium CPT` can be considered best at striking the balance between sensitivity and specificity.



(a) Absolute value results **(b)** Results relative to `No PT`

Figure 4.3: Results from fine-tuning pre-trained models and a model without pre-training on the multimodal dataset with $k_c = 1$ and $p_g = 0.5$.

Figure 4.4 shows sensitivity for aminoglycosides and fluoroquinolones, with a striking difference in performance on AMK between pre-trained models and `No PT`. Overall, the pre-trained models outperform `No PT` in capturing resistance, but there are no clear trends regarding which pre-training strategy yields the best performance.

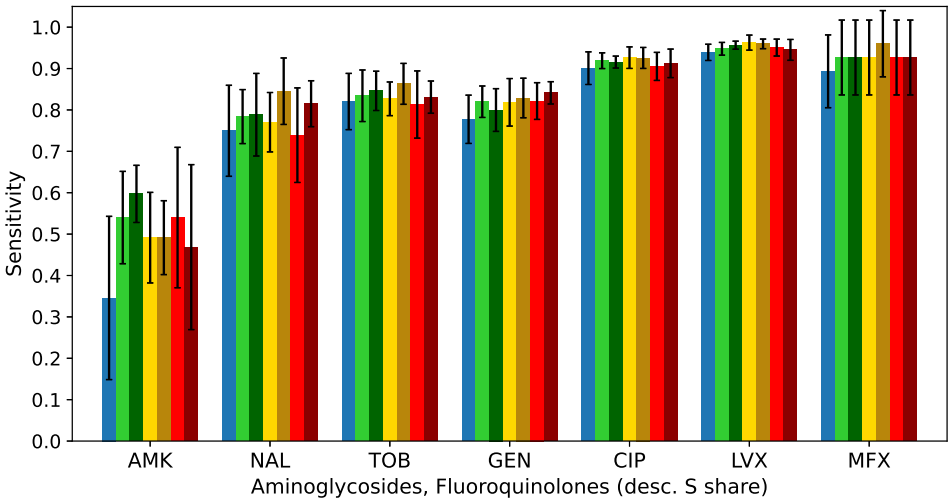
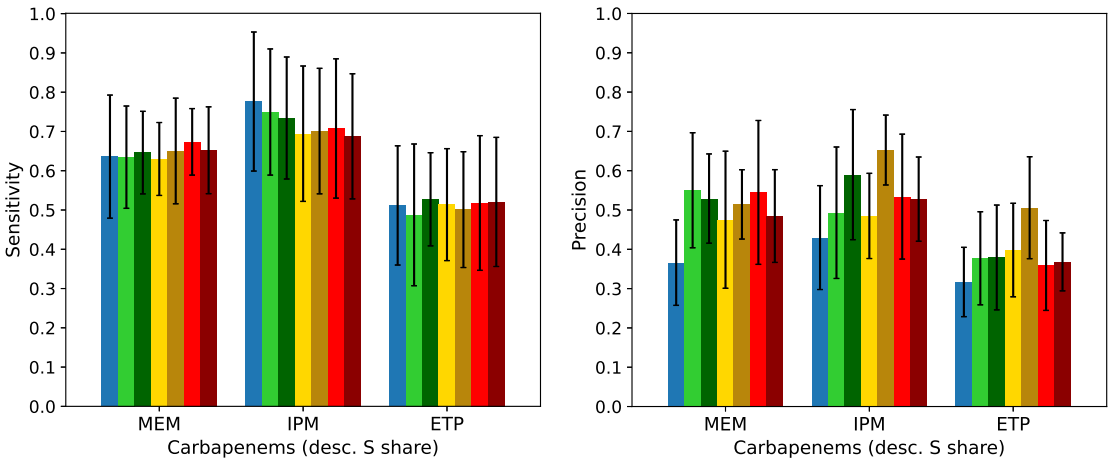


Figure 4.4: Results from fine-tuning with $k_c = 1$ broken down by antibiotic and visualized for aminoglycosides and fluoruoquinolones.

Performance on carbapenems is visualized in Figure 4.5, with sensitivity in panel a) and precision, the accuracy of positive predictions, in panel b). Compared to the random masking task, `No PT` is closer in performance, on average, to the pre-trained models with regards to capturing resistance in carbapenems. However, the precision results clearly show that `No PT` carries an associated cost of low precision and is, in turn, worse at capturing susceptibility. This could explain the larger gap in specificity observed in Figure 4.3, given the high prevalence of isolates susceptible to carbapenems in the data.



(a) Sensitivity **(b)** Precision

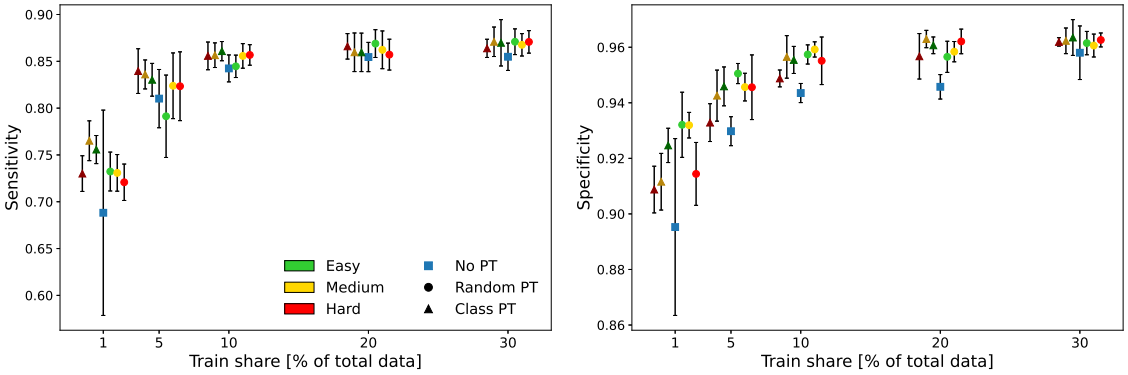
Figure 4.5: Trade-offs in sensitivity and precision on carbapenems for models fine-tuned with $k_c = 1$.

4.1.3 Limited training data

To investigate how model performance is impacted by the size of training data, and if this impact varies depending on the pre-training strategy, the pre-trained models and a model without pre-training were fine-tuned on training sets of reduced size. The reduced datasets will be referred to by their share of the full dataset of 6,494 isolates. Results in Sections 4.1.1 and 4.1.2 act as performance baselines with the models trained on all available training data, i.e. 80% of the dataset.

Random masking

Figure 4.6 shows results from fine-tuning with $p_{ab} = 0.75$ on reduced training sets, with sensitivity in panel a) and specificity in panel b). As expected, average performance is reduced when training data is reduced, though the performance reaches levels comparative to those in Figure 4.1a at 20% train share. The performance is reduced for both sensitivity and specificity, though the performance drop in sensitivity is larger. The model without pre-training shows a trend of considerably worse performance, on average, than pre-trained models in capturing susceptibility at all train shares until 30%, where differences are comparable to those in Figure 4.1b. As expected, the variations in model performance between folds are high and decrease



(a) Sensitivity **(b)** Specificity

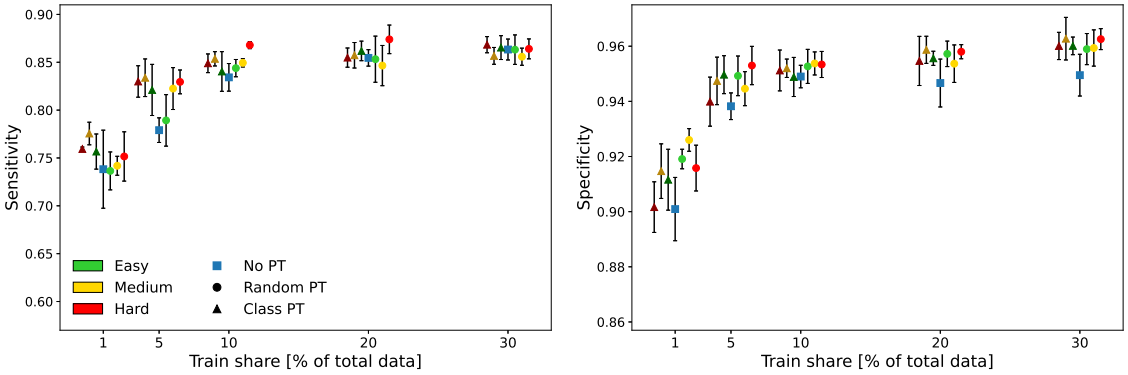
Figure 4.6: Performance of pre-trained models and a model without pre-training after fine-tuning with $p_{\text{ab}} = 0.75$ on reduced train sets as a percentage of total data. Models are colored according to difficulty and masking method during fine-tuning. RPT and CPT are further distinguished by marker type, with a blue square indicating `No PT`, triangles indicating CPT models, and circles indicating RPT models.

as the train share increases. There is a notably higher variation in performance for `No PT` at 1% train share compared to the pre-trained models. This could be due to technical aspects related to the model being randomly initialized and only trained on ~60 isolates, though the model was trained until convergence and met the early stopping criteria for all folds. That the performance drop is larger for sensitivity than specificity is expected, since some antibiotics may only be present as susceptible in the reduced training sets. A model without pre-training would have no reason to predict any other label. However, the pre-trained models will have encountered both labels for all antibiotics during pre-training. This may partially explain why pre-trained models perform better than `No PT` at low train shares.

Class masking

Figure 4.7 shows results from fine-tuning with $k_{\text{c}} = 1$ on reduced training sets. As with performance when training on the whole training set, the performance on the cross-class inference task with reduced training data is comparable, but slightly lower, compared to that on the random masking task. Pre-trained models, and `Medium CPT` in particular, perform better than `No PT` at the lower train shares. However, the trade-off between sensitivity and specificity is apparent, especially at 20% and 30% train share, where `No PT` performs similarly to the pre-trained models in sensitivity, but worse in specificity.

The results in Sections 4.1.1 and 4.1.2 show that the model architecture and training strategies described in Chapter 3 yield good performance on clinically relevant tasks and that the pre-training on large-scale unimodal data improves performance on downstream tasks. However, there are no clear indications on which combination of masking strategy and difficulty in pre-training that yields the best results. The results in Figure 4.1 indicate that models pre-trained at higher difficulty perform better at the random masking task, though no clear pattern is observed on the class



(a) Sensitivity **(b)** Specificity

Figure 4.7: Performance of pre-trained models and a model without pre-training after fine-tuning with $k_c = 1$ on reduced train sets as a percentage of total data.

masking task. Results in Figures 4.6 and 4.7 indicate that CPT models are better at capturing resistance at reduced train share, though at a cost of slightly lower specificity.

When interpreting the results on how pre-training impacts performance, it should be noted that, as was discussed in Section 3.1.2, the two data sources and, in turn, the two datasets, are not aligned in the sense that they differ geographically and in the distributions of some antibiotics. Furthermore, TESSy is a surveillance effort with clinical data, while NCBI gathers samples for research purposes and includes both clinical and environmental samples. If unimodal and multimodal data sources were more aligned, the importance of pre-training might increase. Another drawback in the data is the size of the multimodal dataset. However, it reflects an overall lack of availability of this type of data. A data collection effort that applies both AST testing and genome sequencing on clinical isolates could improve the performance of a multimodal model utilizing the proposed framework. Finally, it should be noted that the lack of structured hyperparameter optimization is a limitation of this work.

4.2 Effect of including genotypes

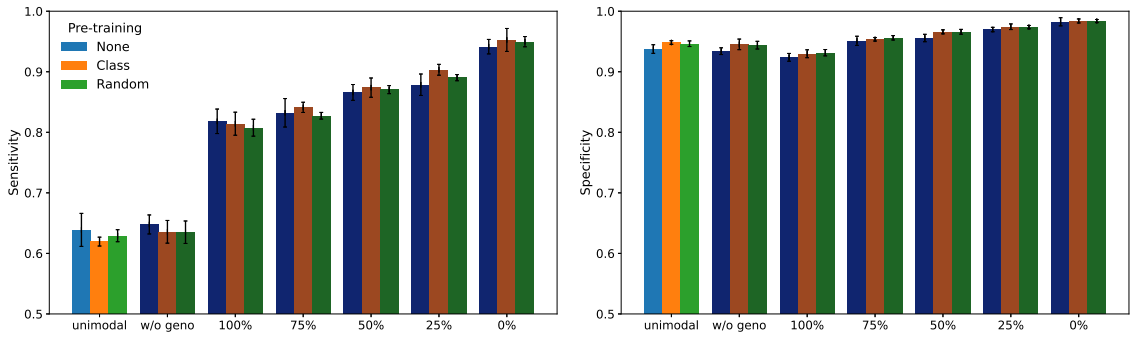
At its core, this project aimed to devise a model that best utilizes available clinical data to predict antibiotic resistance. To achieve this, the approach of using data fusion of metadata and genotype with phenotype was introduced and the results presented in Section 4.1 suggest that this approach yields a model that performs well at this task. In those experiments, genotypes were randomly masked with probability $p_g = 0.5$. To investigate the effect of genotype data availability on performance, the best-performing CPT and RPT models on the class inference task (see Figure 4.3) as well as a model without pre-training, were fine-tuned on that same task at varying genotype masking levels. Additionally, unimodal phenotype models were included to isolate the effect of genotype pre-training on fine-tuning performance. The best-performing unimodal and multimodal models are presented in Table 4.1.

Table 4.1: Pre-training difficulties yielding the best RPT and CPT unimodal and multimodal models on the $k_c = 1$ fine-tuning task based on average F_1.

	Class PT	Random PT
Multimodal	Medium	Easy
Unimodal	Hard	Easy

4.2.1 Multimodality and genotype data availability

Figure 4.8 presents the experiment results for the unimodal and multimodal models at varying levels of available genotype data. When the multimodal models are fed unimodal data, they perform very similarly to their unimodal counterparts. When the multimodal model is fed genotype data masked with $p_g = 100\%$, a large performance gain of over 20 percentage points in sensitivity can be observed, along with a comparatively modest drop of a few percentage points in specificity. This indicates that the model becomes increasingly confident and accurate in making predictions of resistance. As the availability of genotype data increases, the initial trade-off between sensitivity and specificity vanishes. The gains are larger for the pre-trained models than `No PT`, with `Medium CPT` performing best in cases $p_g \leq 75\%$. There is a significant leap in sensitivity to ~95% for all models at $p_g = 0$.



(a) Sensitivity **(b)** Specificity

Figure 4.8: Performance of selected unimodal and multimodal models by varying levels of available genotype information. The bright-colored bars represent unimodal models, while dark-colored bars represent multimodal models. The label "w/o geno" represents the case of multimodal models fed a dataset without genotypes, while the percentages are probabilities of genotype masking. $p_g = 100\%$ entails genotype sequences composed of only `GENE_MASK` tokens.

The observation that the multimodal model performs similarly to the unimodal model when fed unimodal data implies that the multimodal models pre-trained on both the genotype and phenotype tasks are not hampered in their ability to make phenotype predictions on unimodal data. Since the model without pre-training has no way of inferring which specific genotypes lie behind the wall of `GENE_MASK` tokens at $p_g = 100\%$, the performance gains for `No PT` when adding fully masked genotypes can be interpreted as a result of the model learning to see patterns in the number of genotypes and the resistance of antibiotics that is indeed expected to be present

in the data. This is further implied by comparing those gains with the relatively modest gains in performance at $p_g = 75\%$. Whether the pre-trained models achieve their performance gains in the same way is not certain, but implied. Further experimentation and analysis investigating how the multimodal model incorporates the genotype data to make phenotype predictions is needed to fully understand these performance gains.

The performance gains observed by introducing genotype information are not expected to be equally distributed across antibiotics, and performance was broken down on an antibiotic level and pair-wise visualized for a selection of antibiotics and masking levels.

4.2.2 Antibiotic-level impacts of introducing fully masked genotypes

First, to investigate the impact of introducing any genotype information, sensitivity results from multimodal models trained and evaluated on datasets without genotypes versus with 100% masked genotypes are presented in Figure 4.9 for aminoglycosides, fluoroquinolones and AMP, the sole penicillin in the data. These antibiotic classes saw the largest differences in performance, with the aminoglycoside GEN seeing the largest performance gain of almost 60 percentage points. Large gains are also observed in the aminoglycoside TOB, but not as large in AMK, though the variation decreases. For the fluoroquinolones, results are mixed, with performance drops for NAL and MFX and gains for LVX and CIP.

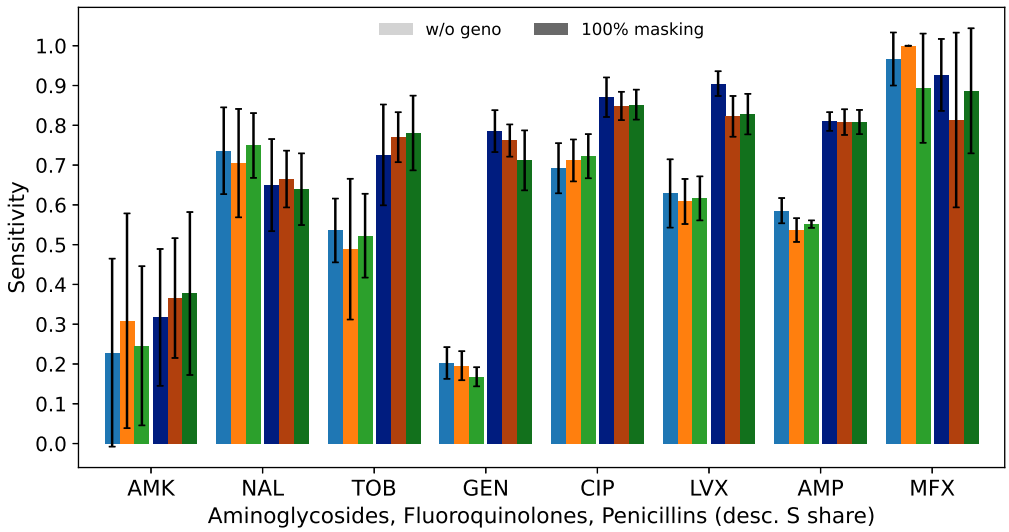


Figure 4.9: Comparison of sensitivity results between the multimodal models fine-tuned with genotype information removed (bright colors) and with all genotypes masked (dark colors). Blue bars represent `No PT` while orange and green colors represent CPT and RPT models, respectively.

Figure 4.10 shows the same comparison for specificity, and the trade-off between the two metrics is observed for most antibiotics. The gains in specificity for NAL

and MFX could explain their drops in sensitivity. The low specificity on MFX can be explained by its imbalanced distribution towards resistance and overall low prevalence in the data. However, the introduction of genotype information allows the model to achieve a better balance between the labels. Indeed, that can be said for all antibiotics.

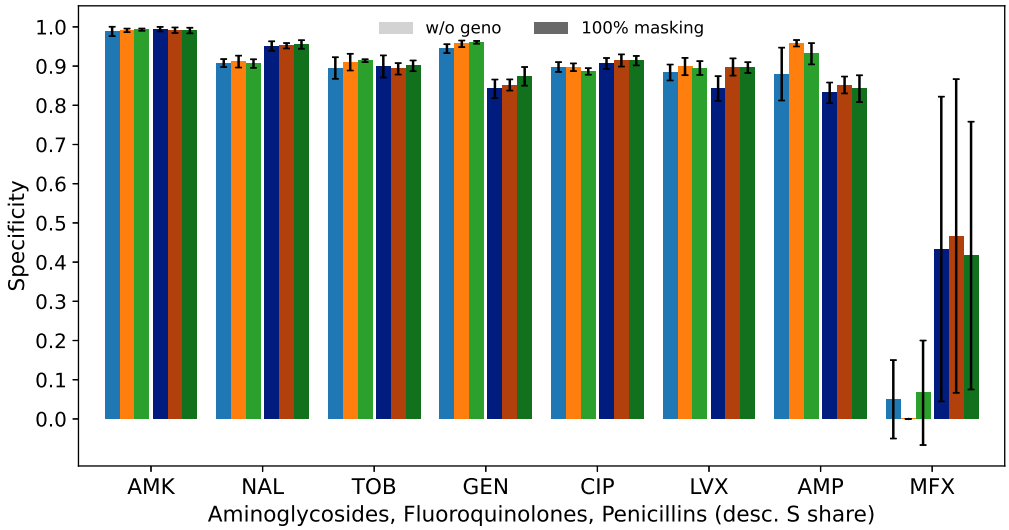


Figure 4.10: Comparison of specificity results between the multimodal models fine-tuned with genotype information removed and with all genotypes masked.

4.2.3 Antibiotic-level impacts of specifying the genotypes

Next, the models were compared when trained and evaluated on datasets with fully masked genotypes versus all genotype information available. Figure 4.11 shows sensitivity for aminoglycosides, fluoroquinolones, and penicillins. There are performance gains across all the antibiotics, and `Medium CPT` reaches ~90% sensitivity on all the antibiotics except AMK. The largest gain appears to be for NAL. Given the results in Figure 4.9, the specific genotypes appear to be more important in informing resistance predictions on NAL compared to other antibiotics.

Figure 4.12 shows the results for specificity, and notably, there are no trade-offs in reduced specificity when specifying the genotypes. This suggests that the model becomes adept at connecting specific genotypes to labels of corresponding antibiotics and that it can use their presence and implied absence to make more accurate predictions. The large gain in specificity for MFX can be explained by its imbalance towards resistance in the fine-tuning data, meaning susceptibility is rare and harder to capture. The results indicate that specifying the genotypes allows the model to identify genotypes that lead to MFX resistance, and that their implied absence leads to MFX susceptibility.

As seen in earlier results, such as Figure 4.5, carbapenems are among the hardest antibiotics in the data to predict, but specifying genotypes improves their performance. Figure 4.13 compares the sensitivity and specificity for carbapenems, and

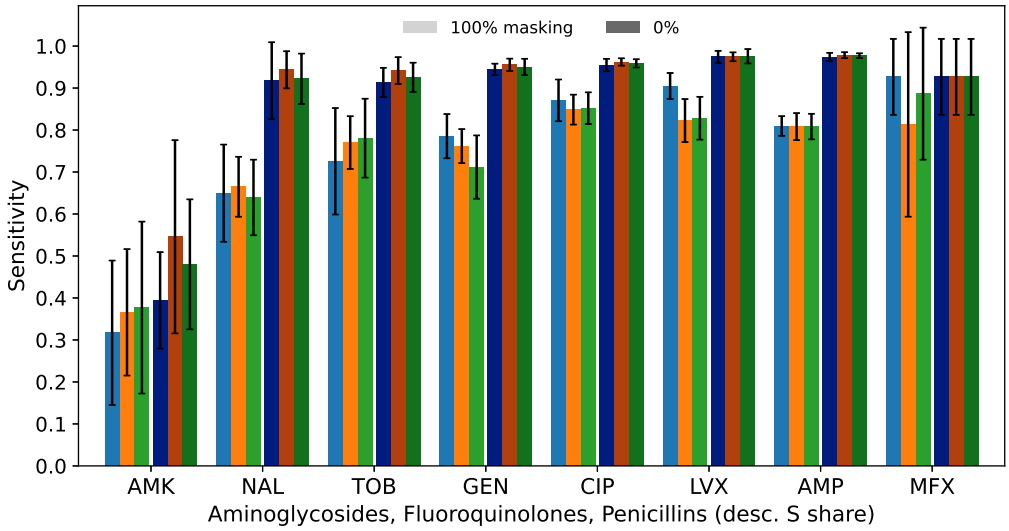


Figure 4.11: Comparison of sensitivity results between the multimodal models fine-tuned with all genotypes masked (bright colors) and all genotypes specified (dark colors).

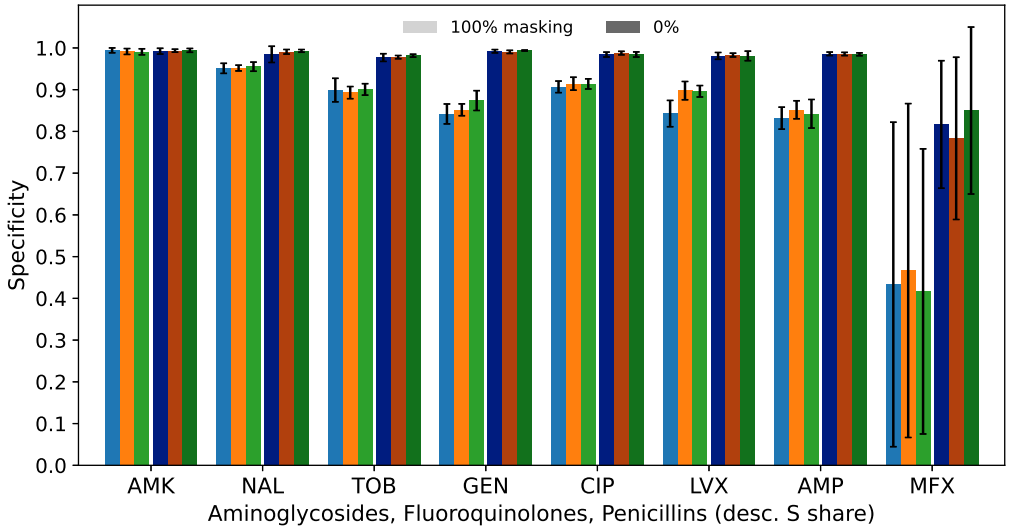
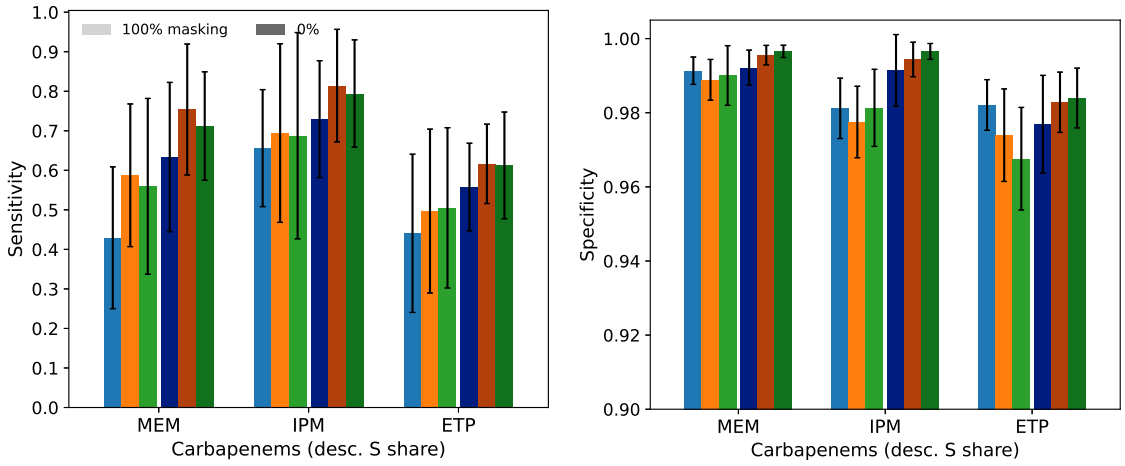


Figure 4.12: Comparison of specificity results between the multimodal models fine-tuned with all genotypes masked (bright colors) and all genotypes specified (dark colors).

considering their extreme label imbalance, the performance is high, with `Medium CPT` capturing ~80% of resistant isolates for MEM and IPM. Importantly, the performance gains in sensitivity for the pre-trained models do not come at the cost of reduced specificity. The average specificity is increased to >99% for MEM and IPM. This indicates that specific genotypes in the data enable the model to better separate the labels for these highly imbalanced antibiotics. While the overall performance levels are not as high for ETP, the absolute gains in performance are comparable, and the variation is reduced. Notably, the average specificity on ETP is increased for the pre-trained models but decreased for `No PT` as the genotypes are specifcied.



(a) Sensitivity **(b)** Specificity

Figure 4.13: Performance comparison on carbapenems between models fine-tuned on datasets with genotype information masked and with all genotypes specified.

The results from this section motivate using a model capable of interpreting genotype data when making antibiotic resistance predictions. Aminoglycosides and AMP benefit most from genotype data, while gains for fluoroquinolones were modest and mixed. Notable performance gains were also observed for carbapenems in Figure 4.13, though only when antibiotics were unmasked. Cephalosporins were excluded from the figures to highlight the most informative results and improve interpretability, but they also saw performance gains from increased genotype information. Section C.2 contains the above results for all antibiotics.

The results in Figure 4.8 show that pre-trained models, and `Medium CPT` in particular, gain more in performance than `No PT` as more genotype information is made available. This suggests that pre-training the model on unimodal data allows it to better utilize genotypes to improve predictions from multimodal data. A potential reason for this is that the MLM task in pre-training allows modeling of the co-occurrence of genes and that the pre-trained models may therefore utilize partial genotype availability better than a model without pre-training that is fine-tuned only on the phenotype prediction task. Together with the results in Section 4.1.3, this suggests that a model pre-trained on large volumes of unimodal genotype and

phenotype data can utilize genotype data to yield high performance even on small fine-tuning datasets. However, the effect of genotype data availability on performance with limited training data should be investigated further. How different pre-training strategies affect how well the model can utilize available genotype data should also be investigated.

Future work could utilize existing data to investigate how the proposed multimodal framework extends to other bacterial species than *E. coli* and how well it could predict antibiotic resistance in multiple species. Certainly, the inclusion of genotype data and a token specifying the species could allow the model to see patterns across species that may improve predictions.

5

Conclusion

The overarching aim of this thesis project was to develop a multimodal transformer that utilizes data fusion of genotype, phenotype and metadata to improve predictions of antibiotic resistance in *E. coli*. By evaluating the model on two clinically relevant tasks, and under the constraint of limited training data, it can be concluded that the framework developed and presented in this work fulfills this aim. Performance is especially high when the model is given access to full genotype information and performance is significantly improved over a unimodal model using only metadata and phenotypes. Future work should focus on further investigation of the role of genotypes under reduced training data and in specific clinically relevant scenarios.

Another aim was to investigate the best way to train the model given the constraints of available data. A pre-training step that seeks to utilize large volumes of unimodal data was developed and evaluated by comparing the fine-tuning performance of models pre-trained with different masking strategies and at different levels of difficulty to a model without pre-training. It can be concluded that pre-training improves performance on the downstream task, particularly when fine-tuned on reduced training data. Furthermore, pre-trained models were better able to utilize genotype data. While separate trends indicate that pre-training with class masking and at higher difficulty improves performance, the results in this work do not single out a specific combination that yields the best performance across tasks and scenarios. Further investigation into pre-training strategies and how they affect performance at different levels of genotype information may unlock these insights. Additionally, improving the quantity of multimodal data that is aligned with unimodal data may yield clearer results on the effect of pre-training.

Bibliography

1. Murray CJL et al. Global burden of bacterial antimicrobial resistance in 2019: a systematic analysis. The Lancet 2022; 399:629–55. DOI: 10.1016/S0140-6736(21)02724-0. Available from: https://www.sciencedirect.com/science/article/pii/S0140673621027240
2. O'Neill J. Tackling drug-resistant infections globally: final report and recommendations. 2016
3. Holmes AH, Moore LS, Sundsfjord A, Steinbakk M, Regmi S, Karkey A, Guerin PJ, and Piddock LJ. Understanding the mechanisms and drivers of antimicrobial resistance. The Lancet 2016; 387:176–87. DOI: 10.1016/s0140-6736(15)00473-0
4. Friedman ND, Temkin E, and Carmeli Y. The negative impact of antibiotic resistance. Clinical Microbiology and Infection 2016; 22:416–22. DOI: 10.1016/j.cmi.2015.12.002
5. Bassetti M, Rello J, Blasi F, Goossens H, Sotgiu G, Tavoschi L, Zasowski EJ, Arber MR, McCool R, Patterson JV, et al. Systematic review of the impact of appropriate versus inappropriate initial antibiotic therapy on outcomes of patients with severe bacterial infections. International journal of antimicrobial agents 2020; 56:106184. DOI: 10.1016/j.ijantimicag.2020.106184
6. Rhee C, Kadri SS, Dekker JP, Danner RL, Chen HC, Fram D, Zhang F, Wang R, Klompas M, Program CPE, et al. Prevalence of antibiotic-resistant pathogens in culture-proven sepsis and outcomes associated with inadequate and broad-spectrum empiric antibiotic use. JAMA network open 2020; 3:e202899–e202899. DOI: 10.1001/jamanetworkopen.2020.2899
7. Falcone M, Bassetti M, Tiseo G, Giordano C, Nencini E, Russo A, Graziano E, Tagliaferri E, Leonildi A, Barnini S, et al. Time to appropriate antibiotic therapy is a predictor of outcome in patients with bloodstream infection caused by KPC-producing Klebsiella pneumoniae. Critical Care 2020; 24:1–12. DOI: 10.1186/s13054-020-2742-9
8. Lodise TP, Kanakamedala H, Hsu WC, and Cai B. Impact of Incremental Delays in Appropriate Therapy on the Outcomes of Hospitalized Adult Patients with Gram-negative Bloodstream Infections:"Every day matters". Pharmacotherapy: The Journal of Human Pharmacology and Drug Therapy 2020; 40:889–901. DOI: 10.1002/phar.2446
9. Gajic I, Kabic J, Kekic D, Jovicevic M, Milenkovic M, Mitic Culafic D, Trudic A, Ranin L, and Opavski N. Antimicrobial susceptibility testing: A com-

prehensive review of currently used methods. Antibiotics 2022; 11:427. DOI: 10.3390%2Fantibiotics11040427

10. Davies J and Davies D. Origins and evolution of antibiotic resistance. Microbiology and molecular biology reviews 2010; 74:417–33. DOI: 10.1128/mmbr.00016-10
11. Carattoli A. Plasmids and the spread of resistance. International journal of medical microbiology 2013; 303:298–304. DOI: 10.1016/j.ijmm.2013.02.001
12. Livermore DM and Pearson A. Antibiotic resistance: location, location, location. Clinical Microbiology and Infection 2007; 13:7–16. DOI: 10.1111/j.1469-0691.2007.01724.x
13. Sakagianni A, Koufopoulou C, Feretzakis G, Kalles D, Verykios VS, and Myrianthefs P. Using Machine Learning to Predict Antimicrobial Resistance—A Literature Review. Antibiotics 2023; 12:452. DOI: 10.3390%2Fantibiotics12030452
14. Vaswani A, Shazeer N, Parmar N, Uszkoreit J, Jones L, Gomez AN, Kaiser Ł, and Polosukhin I. Attention is all you need. Advances in neural information processing systems 2017; 30. DOI: 10.48550/arXiv.1706.03762
15. Lin T, Wang Y, Liu X, and Qiu X. A survey of transformers. AI open 2022; 3:111–32. DOI: 10.1016/j.aiopen.2022.10.001
16. Inda-Díaz JS, Johnning A, Hessel M, Sjöberg A, Lokrantz A, Helldal L, Jirstrand M, Svensson L, and Kristiansson E. Prediction of antibiotic resistance at the patient level using deep learning. bioRxiv 2024. DOI: 10.1101/2023.05.09.539832
17. Singh A, Shannon CP, Gautier B, Rohart F, Vacher M, Tebbutt SJ, and Lê Cao KA. DIABLO: an integrative approach for identifying key molecular drivers from multi-omics assays. Bioinformatics 2019; 35:3055–62. DOI: 10.1093/bioinformatics/bty1054
18. Hutchings MI, Truman AW, and Wilkinson B. Antibiotics: past, present and future. Current opinion in microbiology 2019; 51:72–80. DOI: 10.1016/j.mib.2019.10.008
19. Lima LM, Silva BNM da, Barbosa G, and Barreiro EJ. β-lactam antibiotics: An overview from a medicinal chemistry perspective. European journal of medicinal chemistry 2020; 208:112829. DOI: 10.1016/j.ejmech.2020.112829
20. Serio AW, Keepers T, Andrews L, and Krause KM. Aminoglycoside revival: review of a historically important class of antimicrobials undergoing rejuvenation. EcoSal Plus 2018; 8:10–1128. DOI: 10.1128/ecosalplus.esp-0002-2018
21. Hooper DC. Mechanisms of action of antimicrobials: focus on fluoroquinolones. Clinical infectious diseases 2001; 32:S9–S15. DOI: 10.1086/319370
22. D'Costa VM, King CE, Kalan L, Morar M, Sung WW, Schwarz C, Froese D, Zazula G, Calmels F, Debruyne R, et al. Antibiotic resistance is ancient. Nature 2011; 477:457–61. DOI: 10.1038/nature10388
23. Blair JM, Webber MA, Baylay AJ, Ogbolu DO, and Piddock LJ. Molecular mechanisms of antibiotic resistance. Nature reviews microbiology 2015; 13:42–51. DOI: 10.1038/nrmicro3380
24. Zhao R, Yu K, Zhang J, Zhang G, Huang J, Ma L, Deng C, Li X, and Li B. Deciphering the mobility and bacterial hosts of antibiotic resistance genes under

antibiotic selection pressure by metagenomic assembly and binning approaches. Water research 2020; 186:116318. DOI: 10.1016/j.watres.2020.116318
25. Janiesch C, Zschech P, and Heinrich K. Machine learning and deep learning. Electronic Markets 2021; 31:685–95. DOI: 10.1007/s12525-021-00475-2
26. Sarker IH. Machine learning: Algorithms, real-world applications and research directions. SN computer science 2021; 2:160. DOI: 10.1007/s42979-021-00592-x
27. Sarker IH. Deep learning: a comprehensive overview on techniques, taxonomy, applications and research directions. SN Computer Science 2021; 2:420. DOI: 10.1007/s42979-021-00815-1
28. Macukow B. Neural networks–state of art, brief history, basic models and architecture. *Computer Information Systems and Industrial Management: 15th IFIP TC8 International Conference, CISIM 2016, Vilnius, Lithuania, September 14-16, 2016, Proceedings 15*. Springer. 2016 :3–14. DOI: 10.1007/978-3-319-45378-1_1
29. Terven J, Cordova-Esparza DM, Ramirez-Pedraza A, and Chavez-Urbiola EA. Loss functions and metrics in deep learning. A review. arXiv preprint arXiv:2307.02694 2023
30. Masters D and Luschi C. Revisiting small batch training for deep neural networks. arXiv preprint arXiv:1804.07612 2018. DOI: 10.48550/arXiv.1804.07612
31. Devlin J, Chang MW, Lee K, and Toutanova K. Bert: Pre-training of deep bidirectional transformers for language understanding. arXiv preprint arXiv:1810.04805 2018. DOI: 10.48550/arXiv.1810.04805
32. Qiu X, Sun T, Xu Y, Shao Y, Dai N, and Huang X. Pre-trained models for natural language processing: A survey. Science China Technological Sciences 2020; 63:1872–97. DOI: 10.1007/s11431-020-1647-3
33. The European Surveillance System. European Center for Disease Prevention and Control (ECDC). Available from: https://www.ecdc.europa.eu/en/publications-data/european-surveillance-system-tessy
34. Pathogen Detection Project. National Center for Biotechnology Information (NCBI). Available from: https://www.ncbi.nlm.nih.gov/pathogens/
35. NCBI Pathogen Detection Project Isolates Browser. *Escherichia coli* dataset (version 4460). Available from: https://ftp.ncbi.nlm.nih.gov/pathogen/Results/Escherichia_coli_Shigella
36. NCBI Pathogen Detection help document. Available from: https://www.ncbi.nlm.nih.gov/pathogens/pathogens_help [Accessed on: 2024 Jul 11]
37. Paszke A, Gross S, Massa F, Lerer A, Bradbury J, Chanan G, Killeen T, Lin Z, Gimelshein N, Antiga L, Desmaison A, Köpf A, Yang E, DeVito Z, Raison M, Tejani A, Chilamkurthy S, Steiner B, Fang L, Bai J, and Chintala S. PyTorch: An Imperative Style, High-Performance Deep Learning Library. 2019. arXiv: 1912.01703 [cs.LG]. Available from: https://arxiv.org/abs/1912.01703

A

Appendix: Antibiotic distributions and selection

The thesis project aimed to train a model for the binary classification of antibiotics as resistant (R) or susceptible (S). First, it assumed the 26 antibiotics in the TESSy dataset. These antibiotics are presented in Table A.1 along with their TESSy abbreviations and class affiliations. The distributions of prevalence and label imbalance are significantly skewed for most antibiotics in both datasets. The prevalence of the 26 antibiotics, i.e. rate of occurrence in the phenotype-containing isolates, is presented for both datasets in Table A.2, along with the frequency of S and R labels. The frequencies are for pre-processed datasets and frequencies for the NCBI dataset are calculated after reducing it to isolate containing antibiotics present in TESSy.

Given these unbalanced distributions, and that the task is binary classification, the exclusion of antibiotics from training and evaluation datasets based on these distributions is merited. Since the main task of the model is classification on the multimodal dataset, the selection of antibiotics focuses on the distributions in the NCBI dataset. Firstly, AMC, OFX, and TZP were excluded based on their absence from the NCBI dataset. Secondly, POL, NOR, DOR, COL, AMX, PIP, and NET were excluded mainly due to their low prevalence in the NCBI dataset, but many are also extremely imbalanced. Lastly, TGC was excluded due to a skewed label imbalance combined with a low prevalence. Other imbalanced antibiotics, such as MEM and IPM were kept despite their label imbalance since they are prevalent enough that several minority examples will occur. How the model handles these antibiotics is of interest to the project. Additionally, some antibiotics with low prevalence, such as OFX and MFX are kept due to their (relatively) balanced label distributions. In total, eleven antibiotics are excluded, leaving fifteen antibiotics included in the datasets.

Table A.1: Antibiotics in the TESSy dataset with their abbreviations and class affiliations.

Antibiotic	Abbreviation	Antibiotic class
Amikacin	AMK	Aminoglycosides
Ampicillin	AMP	Penicillins
Amoxicillin	AMX	Penicillins
Ceftazidime	CAZ	Cephalosporins
Ciprofloxacin	CIP	Fluoroquinolones
Colistin	COL	Polypeptides
Ceftriaxone	CRO	Cephalosporins
Cefotaxime	CTX	Cephalosporins
Doripenem	DOR	Carbapenems
Ertapenem	ETP	Carbapenems
Cefepime	FEP	Cephalosporins
Gentamicin	GEN	Aminoglycosides
Imipenem	IPM	Carbapenems
Levofloxacin	LVX	Fluoroquinolones
Meropenem	MEM	Carbapenems
Moxifloxacin	MFX	Fluoroquinolones
Nalidixic acid	NAL	Fluoroquinolones
Netilmicin	NET	Aminoglycosides
Norfloxacin	NOR	Fluoroquinolones
Piperacillin	PIP	Penicillins
Polymyxin B	POL	Polypeptides
Tigecycline	TGC	Tetracyclines
Tobramycin	TOB	Aminoglycosides
Amoxicillin/Clavulanic acid	AMC	Penicillins
Ofloxacin	OFX	Fluoroquinolones
Piperacillin/Tazobactam	TZP	Penicillins

Table A.2: Antibiotic prevalence rates, f, and the frequencies f_S and f_R of labels S and R for the pre-processed TESSy dataset and the NCBI dataset reduced to the isolates containing antibiotics also present in TESSy.

Dataset	NCBI			TESSy		
f denominator	6,494			1,440,857		
Antibiotic	f	f_{S}	f_{R}	f	f_{S}	f_{R}
AMK	0.5054	0.9851	0.0149	0.4966	0.9893	0.0107
AMP	0.9418	0.5353	0.4647	0.6399	0.4433	0.5567
AMX	0.0052	0.0294	0.9706	0.2356	0.4594	0.5406
CAZ	0.5162	0.7159	0.2841	0.8494	0.9170	0.0830
CIP	0.9526	0.8078	0.1922	0.9199	0.8034	0.1966
COL	0.0571	0.9650	0.0350	0.0722	0.9889	0.0111
CRO	0.8982	0.8073	0.1927	0.2733	0.9087	0.0913
CTX	0.1298	0.1588	0.8412	0.7668	0.8996	0.1004
DOR	0.0234	0.7500	0.2500	0.0016	0.9965	0.0035
ETP	0.4960	0.9677	0.0323	0.2446	0.9976	0.0024
FEP	0.5063	0.8140	0.1860	0.2072	0.9054	0.0946
GEN	0.9358	0.8703	0.1297	0.8633	0.9198	0.0802
IPM	0.4382	0.9786	0.0214	0.5094	0.9993	0.0007
LVX	0.4402	0.6973	0.3027	0.1898	0.7780	0.2220
MEM	0.9292	0.9871	0.0129	0.6716	0.9992	0.0008
MFX	0.0088	0.3333	0.6667	0.0638	0.7455	0.2545
NAL	0.5138	0.9314	0.0686	0.0425	0.7328	0.2672
NET	0.0002	0.0000	1.0000	0.0212	0.9538	0.0462
NOR	0.0140	0.8352	0.1648	0.0144	0.7297	0.2703
PIP	0.0011	0.1429	0.8571	0.0532	0.5086	0.4914
POL	0.0005	1.0000	0.0000	0.0012	0.9914	0.0086
TGC	0.0946	0.9902	0.0098	0.1521	0.9951	0.0049
TOB	0.4105	0.9104	0.0896	0.4471	0.9154	0.0846
AMC	-	-	-	0.3450	0.6666	0.3334
OFX	-	-	-	0.0937	0.8219	0.1781
TZP	-	-	-	0.4126	0.9260	0.0740

B

Appendix: Label weights

Table B.1 presents the weights used when fine-tuning the model with weighted BCE loss, as defined in Equation 2.3, along with the label imbalance in the NCBI dataset.

Table B.1: Antibiotic-specific positive-label weights for the weighted BCE loss used during fine-tuning, along with label imbalance in the pre-processed NCBI dataset.

Antibiotic	f_S	f_R	α
AMK	0.985	0.015	0.8
AMP	0.535	0.465	0.55
CAZ	0.716	0.284	0.6
CIP	0.808	0.192	0.7
CRO	0.807	0.193	0.7
CTX	0.159	0.841	0.45
ETP	0.968	0.032	0.8
FEP	0.814	0.186	0.7
GEN	0.870	0.130	0.7
IPM	0.979	0.021	0.8
LVX	0.697	0.303	0.6
MEM	0.987	0.013	0.8
MFX	0.333	0.667	0.55
NAL	0.931	0.069	0.8
TOB	0.910	0.090	0.8

C

Appendix: Complementary figures

C.1 Pre-training strategies

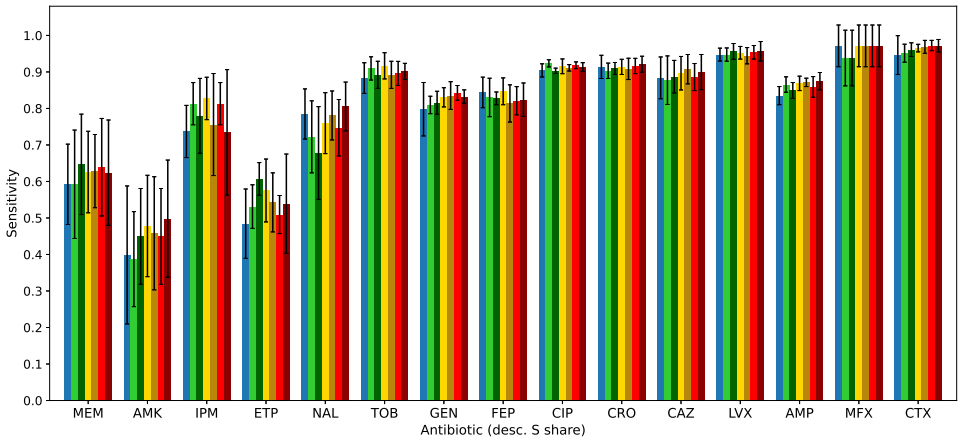


Figure C.1: Sensitivity results from fine-tuning with $p_{\text{ab}} = 0.75$ broken down by antibiotic. Antibiotics are sorted from left to right by decreasing f_{S}, or share of susceptible isolates, meaning the most imbalanced antibiotics are to the left.

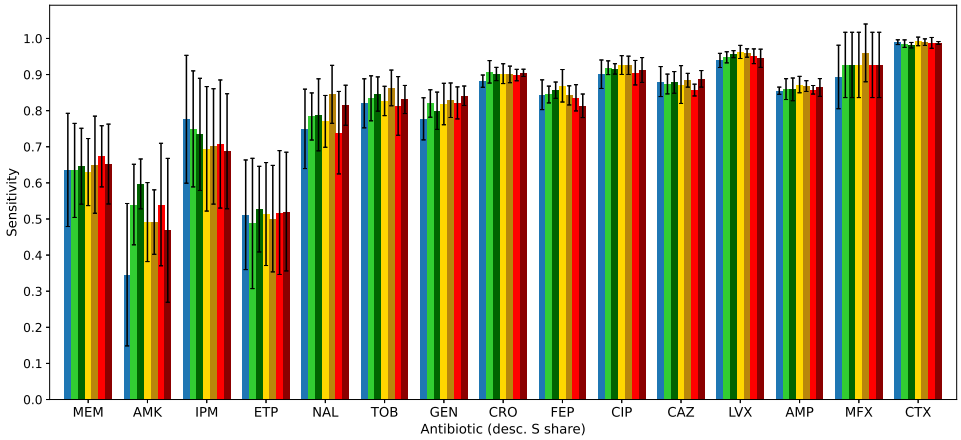


Figure C.2: Sensitivity results from fine-tuning with $k_{\text{c}} = 1$ broken down by antibiotic. Antibiotics are sorted from left to right by decreasing f_{S}, or share of susceptible isolates, meaning the most imbalanced antibiotics are to the left.

C.2 Genotype availability

C.2.1 Including fully masked genotypes

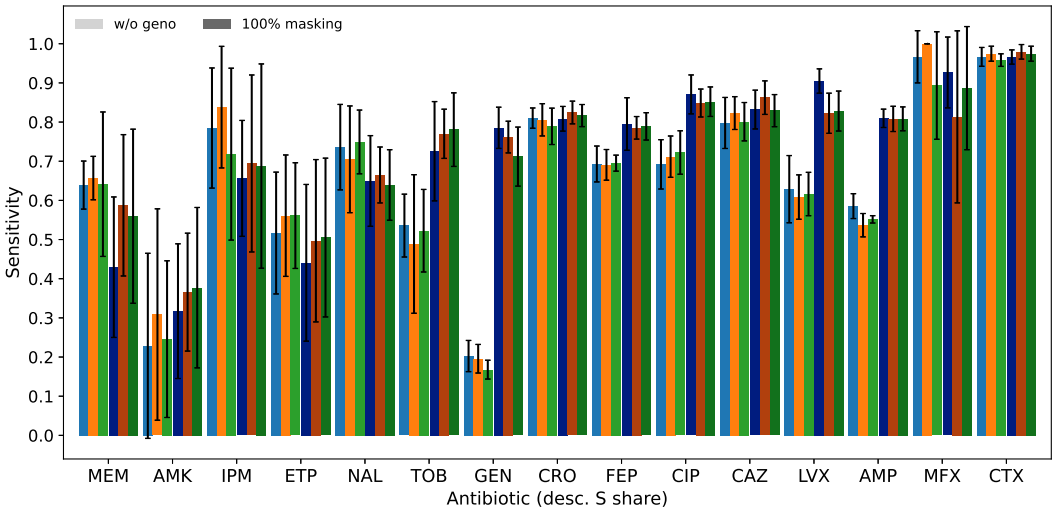


Figure C.3: Comparison of sensitivity results for all antibiotics between the multimodal models fine-tuned with genotype information removed (bright colors) and with all genotypes masked (dark colors).

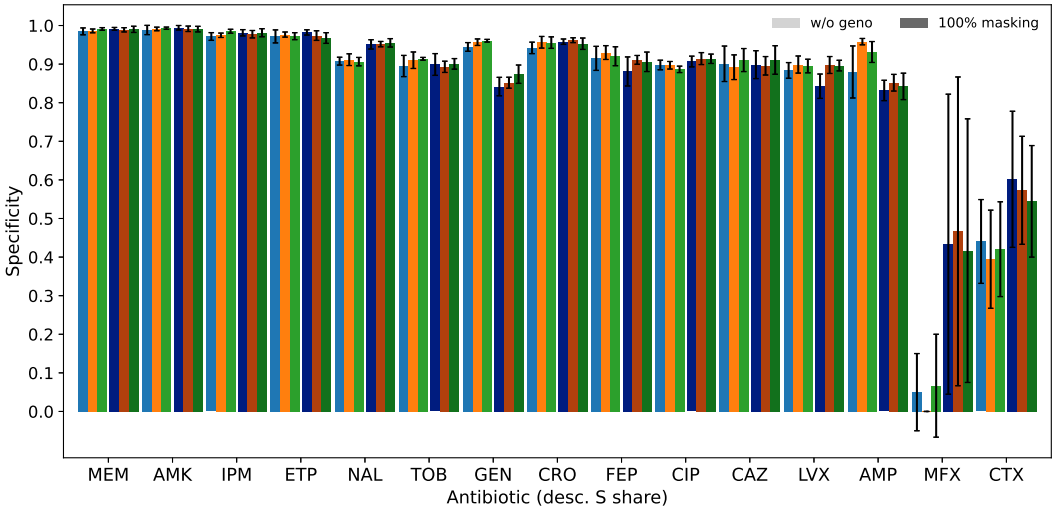


Figure C.4: Comparison of specificity results for all antibiotics between the multimodal models fine-tuned with genotype information removed (bright colors) and with all genotypes masked (dark colors).

C.2.2 Specifying the genotypes

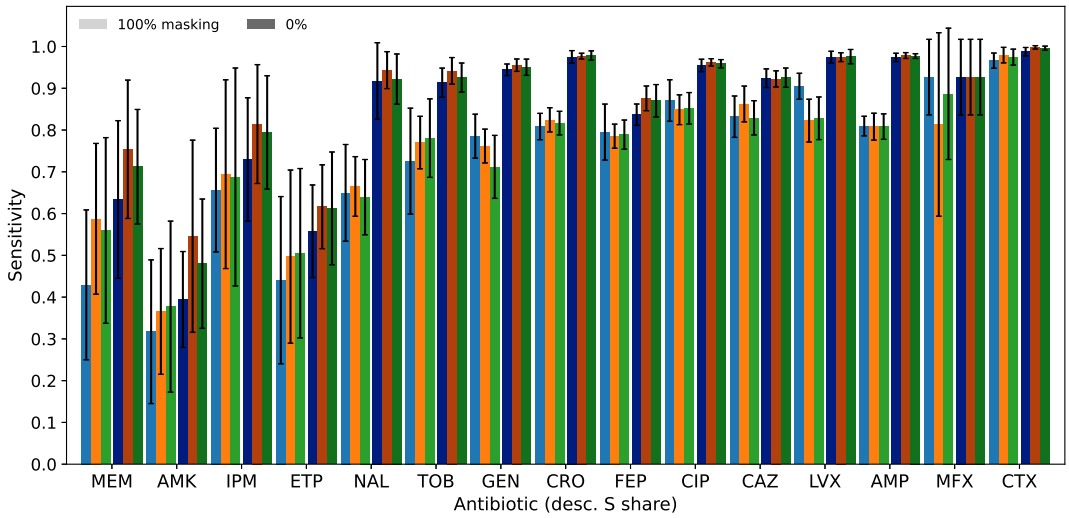


Figure C.5: Comparison of sensitivity results between the multimodal models fine-tuned with all genotypes masked (bright colors) and all genotypes specified (dark colors).

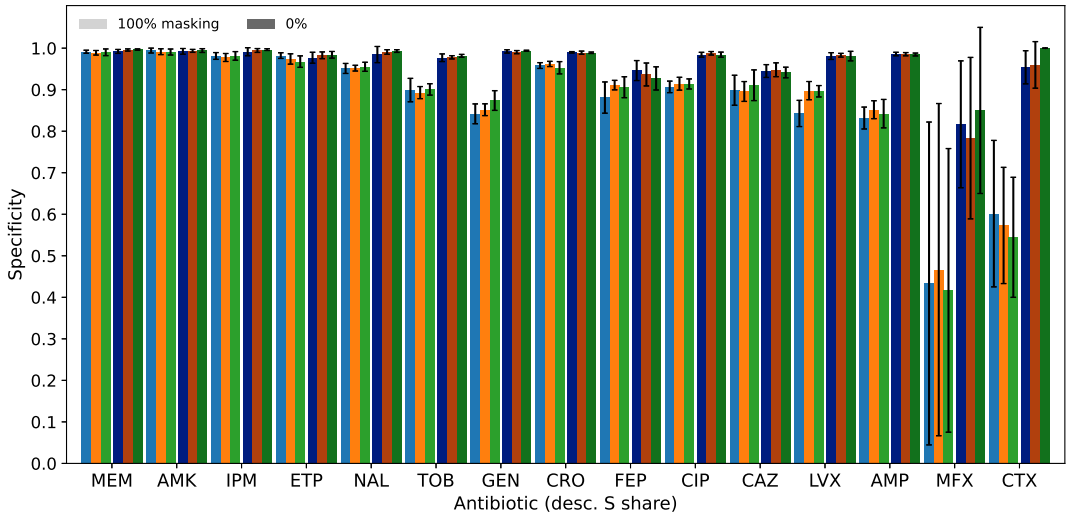


Figure C.6: Comparison of specificity results between the multimodal models fine-tuned with all genotypes masked (bright colors) and all genotypes specified (dark colors).

DEPARTMENT OF MATHEMATICAL SCIENCES
CHALMERS UNIVERSITY OF TECHNOLOGY
Gothenburg, Sweden
www.chalmers.se

CHALMERS
UNIVERSITY OF TECHNOLOGY