Machine Learning Applications and Developments for Swine Industry

By

JEONGHOON KIM
DISSERTATION

Submitted in partial satisfaction of the requirements for the degree of

DOCTOR OF PHILOSOPHY

in

Applied Mathetmatics

in the

OFFICE OF GRADUATE STUDIES

of the

UNIVERSITY OF CALIFORNIA

DAVIS

Approved:

Xin Liu, Chair

Beatriz Martínez-López

Thomas Strohmer

Committee in Charge

2023

Contents

Abstract

This dissertation explores the potential of machine learning in the swine industry, which faces numerous challenges such as disease outbreaks, antimicrobial resistance, and the need for efficient decision-making for optimal production processes. Traditional methods may not be scalable, accurate, or timely enough to meet the demands. Machine learning provides data-driven solutions that enhance decision-making, improve disease prediction, optimize antimicrobial usage, and enhance overall production efficiency. The dissertation consists of six chapters. Chapter 1 presents an introduction and overview of the remaining chapters. Chapters 2, 3, and 4 present machine learning applications to the swine industry. Chapters 5 and 6 present the development and understanding of deep learning models, specifically for novelty detection and synthetic tabular data generation.

Chapters 2, 3, and 4 address real-world problems in the swine industry using machine learning methods with real-world data. Each Chapter focuses on a specific issue: virus classification, antimicrobial resistance (AMR) prediction, and Minimal Inhibitory Concentration (MIC) prediction. Specifically, Chapter 2 aims to classify the Porcine Reproductive and Respiratory Syndrome Virus, a highly infectious disease of pigs, into four different sublineages via amino acid scores from Open Reading Frame 5 gene sequences. Chapter 3 focuses on predicting the future AMR burden of the bacterial pathogen through time series analysis. The goal of Chapter 4 is to predict MIC values for 12 antibiotics using Random Forest based on k-mer counting data processing approach for whole gene information of *Streptococcus suis.*

Chapters 5 and 6 investigate the improvement of training data quality for machine learning applications through the use of deep generative models. These models, such as Generative Adversarial Networks and Autoencoders, are employed for novelty detection and synthetic tabular data generation. Specifically, Chapter 5 proposes a new method based on Adversarial Autoencoders for detecting novelties in multi-modal normality cases. Chapter 6 provides a systematic and comprehensive assessment of how label noise influences synthetic tabular data generation using deep generative model-based synthesizers.

Acknowledgments

I would like to extend my deepest gratitude to my advisor, Professor Xin Liu. Her continual guidance, unwavering encouragement, steadfast support, and immense patience have been integral to my growth as a thorough and critical researcher. The invaluable opportunities she provided have not only broadened my scientific horizon but also honed my problem-solving abilities.

My heartfelt appreciation also goes to my dissertation committee: Professor Beatriz Martínez-López and Thomas Strohmer. Their classes, collaborations, and probing questions have imparted valuable lessons and have always nudged me to view my research from a broader perspective.

This dissertation, undoubtedly, stands as a testament to fruitful collaborations. I consider myself fortunate to have had the opportunity to work with exceptional collaborators. Their contributions have immensely enriched this work and made the journey an enjoyable one.

Furthermore, I am thankful to all the members of the Xin Liu's research group: Avi, Chao, Shahbaz, Taeyoung, Xiaoxiao, Fanyu, Yongshuai, Ziwen, Albara, and Rex. Their dedication, intriguing problem-solving approaches, and stimulating discussions during group meetings have always served as a source of inspiration for me.

Finally, but certainly most importantly, my heartfelt appreciation extends to my parents and older brother. Their unwavering support and unwaning belief in me have been my pillar of strength throughout this journey. This achievement would not have been possible without them standing by me every step of the way.

CHAPTER 1

Introduction

The swine industry, as a critical component of global food production and public health, faces numerous challenges that require innovative approaches for effective management [**7**, **69**]. One such approach is the integration of machine learning techniques, which have gained prominence in recent years for their ability to leverage data-driven methodologies to tackle complex problems [**59**, **62**, **104**]. In this dissertation, we explore the applications of machine learning in the swine industry, addressing diverse perspectives and presenting original research papers that contribute to advancing the field.

The swine industry operates in a dynamic and evolving environment, facing challenges such as disease outbreaks [**19**, **51**, **73**], antimicrobial resistance [**5**, **7**, **128**], and the need for efficient decision-making to optimize production processes [**62**, **116**]. Traditional approaches to addressing these challenges often rely on labor-intensive and time-consuming manual methods, which may not be scalable, accurate, or timely enough to effectively manage the industry's demands [**11**, **120**]. In this regard, machine learning, a subset of artificial intelligence, offers the potential to revolutionize the swine industry by providing data-driven solutions that can enhance decision-making, improve disease detection and prediction, optimize antimicrobial usage, and enhance overall production efficiency.

One of the key reasons why machine learning approaches are crucial in the swine industry is the complexity and variability of the data generated from various sources such as genetics, environment, health records, and production practices. Machine learning algorithms can effectively process and analyze large volumes of data, uncovering hidden patterns, identifying correlations, and extracting valuable insights. These data-driven approaches can provide a better understanding of disease dynamics, identify risk factors, and support evidence-based decision-making for disease prevention, diagnosis, and control. Moreover, machine learning techniques can facilitate the identification of antimicrobial resistance patterns, predict optimal antimicrobial usage, and contribute to the development of sustainable and responsible antibiotic stewardship practices in the swine industry.

In addition, machine learning can enhance the efficiency of swine production processes through predictive modeling, optimization, and automation. For instance, by leveraging machine learning algorithms, swine producers can make data-driven decisions on optimal feed formulations, vaccination strategies, and production management practices. This can lead to improved production outcomes, reduced costs, and enhanced overall operational efficiency. Furthermore, machine learning can enable early detection of disease outbreaks, which can prevent or mitigate potential economic losses and public health risks. This can be achieved through real-time monitoring of various data sources, such as sensor data, animal health records, and environmental data, and using machine learning algorithms to detect abnormal patterns or changes in the data, providing timely alerts to enable prompt intervention.

In conclusion, the swine industry can greatly benefit from the applications of machine learning, which have the potential to revolutionize disease detection and prediction, optimize antimicrobial usage, enhance production efficiency, and support evidence-based decision-making. This dissertation aims to contribute to the growing body of knowledge in the field of machine learning applications in the swine industry through a series of original research papers that address critical challenges and present novel methodologies. Through these efforts, we aim to advance the field and provide valuable insights for the swine industry to effectively manage its complex demands.

The rest of the dissertation has five Chapters falling into two different parts: Chapters 2, 3, and 4 cover machine learning applications in the swine industry using real-world data. Chapters 5 and 6 focus on developing and understanding deep learning models for novelty detection and synthetic tabular data generation, respectively. Below is a brief overview of the chapters.

Overview of Chapter 2

This chapter explores machine learning applications for virus classification. Specifically, we aim to classify strains of Porcine Reproductive and Respiratory Syndrome Virus (PRRSV), a highly infectious disease of pigs, into four clades using amino acid scores based on Open reading frame 5 (ORF5) gene sequence information. This study is of great significance as PRRSV causes reproductive failures in sows and respiratory disease in pigs of all ages, resulting in substantial economic losses in the swine industry worldwide, including the United States (US), where annual losses have been estimated at $664 million [**40**]. We specifically utilized the ORF5 gene as it is known to be a crucial resource for classifying field PRRSV strains [**15**, **53**]. Traditionally, there are two methods that have been widely used in literature for strain classification: Restriction Fragment Length Polymorphism (RFLP) typing and phylogenetic tree analysis. However, both methods have their limitations: RFLP provides quick results with unstable accuracy [**120**], while phylogenetic trees return high accuracy but require excessive computational power [**101**]. To overcome these limitations, we employed machine methods including random forest, k-nearest neighbor, support vector machine, and multilayer perceptron, as they can offer rapid and accurate classification results compared to the traditional approaches. In this study, we used amino acid sequences of ORF5 gene from 1931 field PRRSV strains collected in the US from 2012 to 2020. Phylogenetic analysis was used to label field PRRSV strains into one of four clades: Lineage 5 or three clades in Lineage 1. We measured the accuracy and time consumption of classification, two key metrics, using four machine learning approaches with different sizes of gene sequences. We found that all four algorithms classify a large number of field strains in a very short time ($<$ 2.5 seconds) with very high accuracy ($>$ 0.99 Area Under the curve of the Receiver of Operating Characteristics curve). Furthermore, the random forest approach identified a total of 4 key amino acid positions that were crucial for the classification of field PRRSV strains into four clades. Our findings provide insights for developing a rapid and accurate classification model using genetic information, which can enable real-time or semi-real-time handling of large genome datasets for data-driven decision-making and more timely surveillance.

Overview of Chapter 3

In this Chapter, we aim to predict the future Antimicrobial resistance (AMR) burden of bacterial pathogens via time series analysis. AMR poses significant health and economic challenges in our society. Detecting the emergence and spread of AMR in food animal production is crucial for effective mitigation, but current methods such as Minimum Inhibitory Concentration (MIC) testing using genotypic information can be costly and time-consuming due to bacterial growth rates. To address this issue, we employed time series approaches to predict the future burden of AMR in bacterial pathogens because they use only historical records of AMR without the need for genotypic information. We collected comprehensive pathogen and antimicrobial data from over 600 farms in the United States spanning from 2010 to 2021 to generate AMR time series data. Our prediction focused on five bacterial pathogens, namely *Escherichia coli, Streptococcus suis, Salmonella sp.,* *Pasteurella multocida,* and *Bordetella bronchiseptica.* Among the various models evaluated, Seasonal Auto-Regressive Integrated Moving Average (SARIMA) outperformed five baseline models, including Auto-Regressive Moving Average (ARMA) and Auto-Regressive Integrated Moving Average (ARIMA). Our findings provide valuable tools for predicting the AMR burden not only for the pathogens assessed in this study but also for other bacterial pathogens. This approach has the potential to significantly improve the accuracy and efficiency of AMR surveillance, enabling timely detection and intervention to mitigate the spread of AMR in food animal production. This research contributes to addressing the urgent need for rapid and accurate AMR detection to combat this critical global health challenge.

Overview of Chapter 4

As an extension of the study from Chapter 3, this study aims to predict MIC values explicitly using phenotypic data, i.e., whole genome information of the pathogen. Specifically, 203 *Streptococcus suis* whole genome sequences are utilized for prediction. In this study, the main challenges lie in the data itself such as limited data samples, incomplete feature information, and imbalanced data class. To tackle these issues and process whole genome information data, k-mer counting methods were adopted because 1) it has shown great performances in similar studies [**83**, **110**, **126**], 2) it is sensitive to the genomic features such as nucleotide composition, which is crucial in our situation where the data sample is very limited, and 3) it is scalable by simply adjusting the k-mer size (k value) according to the desired level of resolution and analysis complexity. After processing data, we utilized Random Forest regression to predict phenotypic data using genotypic information. To evaluate our regression results comprehensively, three different metrics were used: Accuracy (ACC), Major Error (ME), and Very Major Error (VME). To be specific, VME indicates the proportion of wrong prediction for resistant samples, i.e., the proportion of the resistant genomes that have been assigned susceptible MICs by the model. Food and Drug Administration (FDA) standards [**27**, **79**] for the VME rates indicate that the lower and upper 95% confidence limits should be 1.5% and 7.5%, respectively. Similarly, the ME indicates the proportion of wrong prediction for susceptible samples, i.e., the proportion of the susceptible genomes that have been assigned resistant MICs by the model. For this metric, FDA standards recommend a ME rate 3%. Our results show competitive outcomes with a very limited data sample size compared to similar studies. With a larger data sample size close to that in similar studies, better outcomes are expected to be achieved.

Overview of Chapter 5

In this study, we aim for novelty detection using deep generative models, specifically, Autoencoder (AE) [**30**]. Novelty detection is significant for machine learning applications as novelty detection can help improve the quality of training data for machine learning tasks by identifying and potentially removing outliers or anomalous data points that could negatively impact model performance. In most similar studies, image reconstruction error has been used as a novelty score function. However, image data, high dimensional as it is, contains a lot of different features other than class information which makes models hard to detect novelty data properly. The problem gets even more difficult in multi-modal normality cases. To address this challenge, we propose a new way of measuring novelty scores in multi-modal normality cases using orthogonalized latent space. Specifically, we employ orthogonal low-rank embedding in the latent space to disentangle the features in the latent space using mutual class information. With the orthogonalized latent space, the novelty score is defined by the change of angle in each latent vector. The proposed algorithm was compared to state-of-the-art novelty detection algorithms using Generative Adversarial Networks (GAN) [**30**] such as RaPP [**52**] and OCGAN [**89**], and experimental results show that ours outperforms baseline algorithms. This study can potentially help to detect novelty/outliers in swine data as it is practically possible to acquire noisy information when collecting data such as farm-level sensor data.

Overview of Chapter 6

This Chapter focuses on the evaluation of the impact of label noise on synthetic data generation. This task aids in enhancing training data quality for machine learning tasks by potentially identifying and mitigating the effects of label noise on generated samples. In many practical machine learning-based application cases including the swine industry, data shortage is an issue as real data is not acquired easily due to many reasons such as budget and privacy. Synthetic data has been actively used for various machine learning-based tasks due to its benefits such as massive re-productivity and privacy enhancement compared to using the original data. The quality of the generated synthetic dataset crucially depends on the quality of the original data, which, in practice, is usually, corrupted by label noise. While there have been studies on feature noise, how label noise affects synthetic data generation is under-explored. In this paper, we evaluate the impact of the label noise label on synthetic data generation with a focus on tabular data. One challenge is how to evaluate the quality of synthetic data under label noise. To this end, we design comprehensive experiments to measure the impact of label noise on synthetic data generation in different aspects: synthetic data quality, data utility, and convergence rate for training synthesizers, and machine learning models for downstream tasks. The empirical results cover wide aspects of synthetic data generation under label noise. Along with the study in Chapter 5, this study can help increase performance enhancement by improving data quality for machine learning models.

CHAPTER 2

Applications of Machine Learning for the Classification of Porcine Reproductive and Respiratory Syndrome Virus Sublineages Using Amino Acid Scores of ORF5 Gene

Published in *Frontiers in Veterinary Science* (July 2021).
Edited for this dissertation.

Joint work with:
Kyuyoung Lee, Ruwini Rupasinghe, Beatriz Martínez-López
Department of Medicine and Epidemiology, Center for Animal Disease Modeling and Surveillance (CADMS), School of Veterinary Medicine, University of California, Davis, Davis, CA, United States
{pvmlee, rkrupasinghe, beamartinezlopez}@ucdavis.edu
Shahbaz Rezaei, Xin Liu
Department of Computer Science, University of California, Davis, Davis, CA, United States
{srezaei, xinliu}@ucdavis.edu

2.1. Abstract

Porcine reproductive and respiratory syndrome is an infectious disease of pigs caused by PRRS virus (PRRSV). A modified live-attenuated vaccine has been widely used to control the spread of PRRSV and the classification of field strains is a key for successful control and prevention. Restriction fragment length polymorphism targeting the Open reading frame 5 (ORF5) genes is widely used to classify PRRSV strains but showed unstable accuracy. Phylogenetic analysis is a powerful tool for PRRSV classification with consistent accuracy but it demands large computational power as the number of sequences gets increased. Our study aimed to apply four machine learning (ML) algorithms, random forest, k-nearest neighbor, support vector machine, and multilayer perceptron, to classify field PRRSV strains into four clades using amino acid scores based on ORF5 gene sequence.

Our study used amino acid sequences of the ORF5 gene in 1931 field PRRSV strains collected in the US from 2012 to 2020. Phylogenetic analysis was used to label field PRRSV strains into one of four clades: Lineage 5 or three clades in Lineage 1. We measured the accuracy and time consumption of classification using four ML approaches with different sizes of gene sequences. We found that all four ML algorithms classify a large number of field strains in a very short time ($<$ 2.5 seconds) with very high accuracy ($>$ 0.99 Area under the curve of the Receiver of operating characteristics curve). Furthermore, the random forest approach detects a total of 4 key amino acid positions for the classification of field PRRSV strains into four clades. Our finding will provide an insightful idea to develop a rapid and accurate classification model using genetic information, which also enables us to handle large genome datasets in real time or semi-real time for data-driven decision-making and more timely surveillance.

2.2. Introduction

Porcine reproductive and respiratory syndrome is one of the most important infectious diseases of pigs caused by PRRS virus (PRRSV), an enveloped RNA virus in the genus arterivirus. The virus causes reproductive failures in sows and respiratory disease in pigs of all ages, resulting in significant economic losses in the swine industry worldwide including in the United States of America (US), in which the annual losses have been estimated at $664 million [**40**]. PRRSV strains diverged into multiple lineages globally and two major genotypes were reported in distinct geographical regions: Type 1 PRRSV in Europe and type 2 in North America [**105**]. A modified live-attenuated vaccine (MLV) developed for type 2 PRRSV (e.g. Ingelvac PRRS® MLV by Boehringer Ingelheim Vetmedica, Inc. for lineage 5) has been widely used to control PRRSV in the US Porcine industry for more than 20 years [**78**]. However, the generation and spread of novel strains and/or the virulence reversion of vaccine strains [**56**] have an impact on the efficacy of MLV and consequent spread of PRRSV type 2 in the US swine population. Consequently, the classification of field PRRSV strains played an important role of successful control and prevention measures of PRRSV type 2 in the US using MLV, especially for monitoring the effectiveness of vaccination as well as the development of new vaccines such as vaccine lineage selection. (e.g. Prevacent® by Elanco Inc. for lineage 1, PrimePac™ by Merck, Inc. for lineage 7, Fostera® by Zoetis and Ingelvac ATP by Boehringer Ingelheim Vetmedica, Inc. for lineage 8) [**86**]. The PRRSV genome consists of ten open reading frame (ORF) genes (1a, 1b, 2a, 2b, 3, 4, 5, 5a, 6, 7), and the ORF5 gene encodes the GP5 protein; a hypervariable and immunogenic domain of PRRSV. The genetic information of the ORF5 gene in PRRSV is a key target to classify field PRSSV strains and evaluate the cross-protection induced by MLV [**15**, **53**]. Restriction fragment length polymorphism (RFLP) typing has been widely used to classify field strains due to relatively low experimental cost and short time consumption [**120**]. However, current experimental verification of RFLP typing casted doubt on the stability and accuracy in PRSSV classification considering continuous mutation in the ORF5 gene even among field PRRSV strains with very close genetic relatedness [**11**]. Corresponding to the current use of viral genome sequencing and open-source genomic data repositories, phylogenetic analysis has been increasingly employed to classify PRRSV strains due to consistent accuracy. However, phylogenetic analysis posed a challenge to estimate the phylogeny with a large number of genetic sequences because of the exponential increase of computational power for the calculation of likelihoods of possible combination phylogenies. Machine learning (ML) has been widely used for classification and prediction in computer vision and natural language processing [**107**]. ML is preferred for highly complex classification including multi-dimension and multi-class datasets rather than the regression model because ML has the strength to find the best-fit decision boundaries among discrete values and output class labels. A variety of ML algorithms have been developed for classification. Specifically, four ML algorithms, random forest (RF), support vector machine (SVM), k-nearest neighbors (KNN), and multilayer perceptron (MLP) are widely applied because of high accuracy, applicability, and adaptability [**22**, **108**]. Despite of the great potential, ML has not been easily applied for the classification using genetic information such as DNA, RNA or amino acid sequences. Genome data are coded in long strings of alphabetic letters describing unique biochemical components (e.g., Adenine (A), Guanine (G), Cytosine (C), and Thymine (T)). Therefore, a simple transformation of the genome data in the numeric form possibly leads to significant information loss. Atchley WR et al (2005) presented the approach to transform the amino acid sequence into five numerical scores describing physicochemical properties (e.g. polarity, secondary structure, molecular volume, codon diversity, and electrostatic charge) [**4**]. The amino acid score provided availability to use ML algorithms for prediction and classification of phenotypic characteristics based on genetic information [**92**]. The present study aimed to classify

US field PRRSV stains into four clades using four ML approaches based on amino acid scores of the ORF5 gene. Second, we will also detect key amino acid positions for the classification. To the best of our knowledge, our work is the first attempt to apply four ML algorithms for the classification of field PRRSV strains. Our study will provide an insightful idea to develop a rapid and accurate classification model using genetic information of infectious pathogens, which also enables us to handle large datasets in real-time or semi-real time for data-driven decision-making and more timely surveillance or intervention strategies.

2.3. Materials and methods

2.3.1. Data collection and phylogenetic analysis. We collected ORF5 genome sequences and RFLP types of 1931 field PRRSV strains isolated from 328 porcine premises managed by two US pork production systems from 2012 to 2020. Multiple sequence comparison by log-expectation was used to align ORF5 nucleotide sequences on AliView [Version 1.26] [**57**]. Homogeneity over alignment was evaluated, and common almost-pure-gap sites and the last three sites of stop codon were removed. The best-fit nucleotide substitution model and the among site rate variation were determined by ModelFinder [**46**] based on the Bayesian information criterion (BIC). The best-fit model including among site rate variation and the partition scheme corresponding to codon positions were used to estimate the phylogeny of the 1931 ORF5 gene through the maximum likelihood approach on IQ-TREE multicore [Version 2.1.1] in CIPRES Science Gateway [Version 3.3] [**80**]. Bootstrap values were assessed using the ultrafast bootstrap approximation method with 5000 replicates. The phylogenetic tree was visualized by the interactive Tree of Life (iTOL) tool [**60**]. All 1931 ORF5 gene sequences of US field PRRSV samples were labeled into one of four clades: Lineage 5 (L5 clade) or three clades in Lineage 1 (L1A clade: Sublineage 1.5, L1B clade: Sublineage 1.6, and L1C clade: Sublineages 1.7, 1.8 and 1.9) based on the topology of phylogeny with high bootstrap values (>95%) according to the global PRRSV classification systems [**106**]

2.3.2. Data transformation for the classification using four ML algorithms. The nucleotide alignment of the ORF5 gene of 1931 PRRSV samples with 600 nucleotide base pairs were converted into the alignment of amino acid sequences with 200 amino acids long based on genetic code using AliView. Each amino acid sequence transformed to a 5×1 vector including five numerical

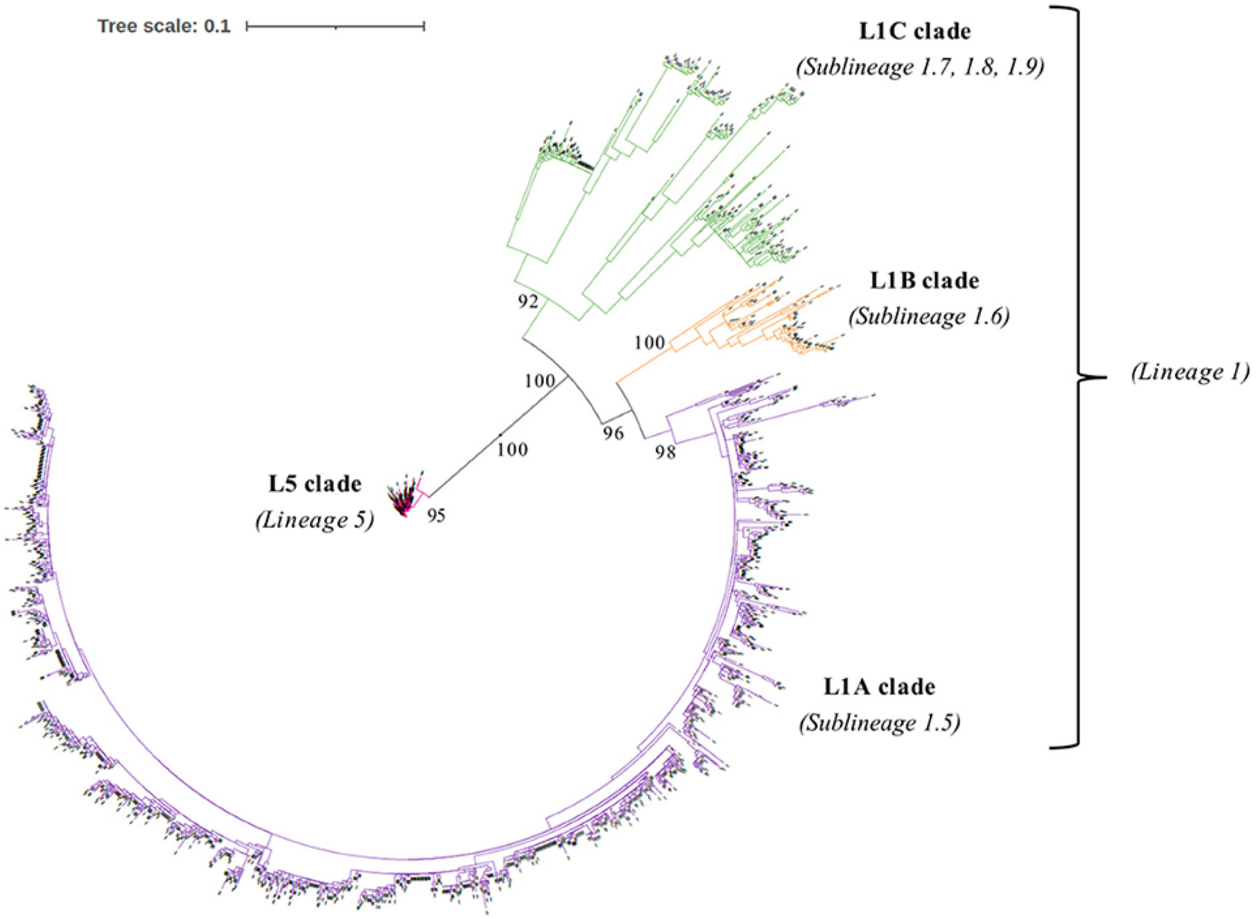


Figure 1: The phylogeny of 1936 Porcine reproductive and respiratory syndrome virus (PPSSRV) field strains estimated by maximum likelihood approach based on the nucleotide sequences of open reading frame 5 (ORF5) gene.

scores of physiochemical amino acid properties (e.g. polarity, secondary structure, molecular volume, codon diversity, and electrostatic charge) [**4**]. The 200 (amino acid sequences) × 5 (numeric score) matrix of each PRRSV ORF5 gene was changed to the 1000×1 matrix for technical convenience. Finally, we built 1931 matrices with 1000 features (1000x1) of PRRSV ORF5 gene sequence for the classification into four clades by four machine learning algorithms (Figure 2). The distribution of 1000 multivariate features and their clusters by four clades were visualized in two-dimension by principal component analysis (PCA).

2.3.3. Machine learning algorithms. Our study used four ML algorithms, random forest (RF), support vector machine (SVM), k-nearest neighbors (KNN), and multilayer perceptron (MLP),

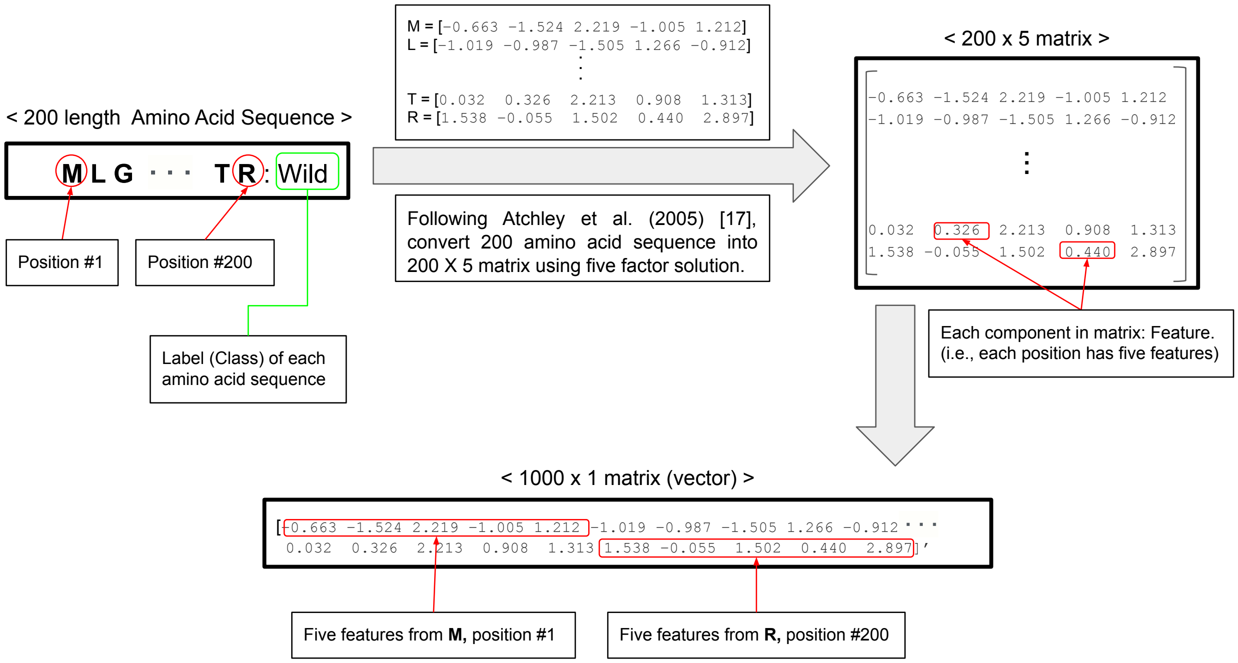

< 200 length Amino Acid Sequence >
M L G · · · T R : Wild
Position #1
Position #200
Label (Class) of each amino acid sequence
M = [−0.663 −1.524 2.219 −1.005 1.212]
L = [−1.019 −0.987 −1.505 1.266 −0.912]
T = [0.032 0.326 2.213 0.908 1.313]
R = [1.538 −0.055 1.502 0.440 2.897]
Following Atchley et al. (2005) [17], convert 200 amino acid sequence into 200 X 5 matrix using five factor solution.
< 200 x 5 matrix >
−0.663 −1.524 2.219 −1.005 1.212
−1.019 −0.987 −1.505 1.266 −0.912
0.032 0.326 2.213 0.908 1.313
1.538 −0.055 1.502 0.440 2.897
Each component in matrix: Feature. (i.e., each position has five features)
< 1000 x 1 matrix (vector) >
[−0.663 −1.524 2.219 −1.005 1.212 −1.019 −0.987 −1.505 1.266 −0.912 · · ·
0.032 0.326 2.213 0.908 1.313 1.538 −0.055 1.502 0.440 2.897]'
Five features from M, position #1
Five features from R, position #200

Figure 2: Flow chart of data preprocessing from an amino acid sequence into five numeric scores.

to classify field PRRSV strains among four clades based on the five amino acid score of ORF5 gene. Additionally, information on theoretical time complexity using big-O-notation were provided.

2.3.3.1. *Random Forest (RF).* RF is a supervised ML algorithm that is used for both classification and regression [**9**]. RF creates several decision trees using training samples and obtains a class prediction from each of the decision trees, and it outputs the final class by majority voting at inference time to achieve high accuracy. RF also provides the importance scores in classification for all features (i.e., 1000 attributes for each sequence), which indicates the importance of each feature for RF classification, by using entropy and information gain. The importance score is the value between 0 to 1; the greater this value is, the more important in classification corresponding feature is. The information on importance scores in classification for all features was used in feature selection in experiments. The training time complexity of RF is O(n*log(n)*d*k) where k is the number of the decision tree, n is the number of samples, and d is the number of dimensions (features) [**65**].

2.3.3.2. *Support vector machine (SVM).* SVM is one of the supervised ML algorithms finding decision boundaries, so-called “a hyperplane”, for the classification of data points in N-dimensional space (e.g. N variable) [**16**]. SVM determines one hyperplane with the maximum margin among many possible hyperplanes based on the maximum distance between the hyperplane and data points in both classes. For data with non-linear and complex features, a non-linear kernel (e.g., polynomial and Radial Based Function) is often used to map the input into a high-dimensional feature space. However, the present study achieved a good classifier without using any non-linear kernel. The training time complexity for SVM is between O(n) and O(n2.3) where n is the number of the training sample [**121**].

2.3.3.3. *k-nearest neighbors (KNN).* KNN is one of the supervised ML algorithms using the proximity of data points for classification based on the assumption, “Similar things are near to each other” [**17**]. Proximity is generally defined by a distance function. The distance function finds k neighbor data points in the training set nearest to the input. Then, the majority vote is performed over the label of the k data points to predict the label of the input. Therefore, a higher value of k often makes the model less sensitive to noise at the cost of more computation. The present study used Euclidean distance as a distance function. The k is a hyperparameter of our KNN algorithm. Our data achieved good performance when $k = 5$. The training time complexity of KNN is O(1), and the prediction time complexity is O(k*n*d) where n and d are the numbers of training samples and dimensions (features), respectively.

2.3.3.4. *Multilayer perceptron (MLP).* MLP is one of the fundamental feedforward neural network architectures used for classification [**76**]. MLP uses one or more hidden layers consisting of many nodes between input and output layers. The MLP architecture takes input data, returns some outputs, and improves the accuracy by repeating three steps: each node (1) takes a weighted sum of its inputs on the connected nodes in the previous layer, (2) performs a non-linear operation (called activation function), and (3) passes the output to some connected nodes in the right next layer. The present study used backpropagation for the training of MLP to obtain optimal weights and bias [**55**]. To reach an approximated solution, the tough time complexity of MLP is O(E*n*d*N) where E is the number of epochs, n is the number of training samples, d is the number of dimensions (features), and N is the number of neurons (nodes) in the architecture.

2.3.4. Evaluation of classification into four clades using four ML algorithms by accuracy and time consumption.

2.3.4.1. *RF classification and detection of key amino acid positions.* We employed RF to classify 1931 US field PRRSV samples into four clades based on the matrix of 1000 features. RF returned the importance scores for all 1,000 features. The importance score for one amino acid position was the sum of five importance scores for five features describing the physiochemical properties of one amino acid in the position. For example, the importance score of the amino acid position 1 was the sum of the five importance scores of feature 1 to 5 in the matrix of 1000 features because feature 1 to 5 are the five physiochemical amino acid properties of features of the position 1 amino acid.

2.3.4.2. *Classification accuracy and time consumption of four ML algorithms by the number of amino acid sequences.* We evaluated the accuracy and time consumption for field PRRSV classification into four clades using four ML algorithms (RF, SVM, KNN, and MLP). We first measured the accuracy and time consumption of four ML algorithms using 200 amino acid positions. To evaluate how the number of amino acid sequences affected the performance of four ML algorithms, we measured the accuracy and time consumption of four ML algorithms using one amino acid position with the highest RF importance score and sequentially added amino acids from the position with the second highest RF importance score. The 10-fold cross-validation was assigned, and training and test data were randomly split in each run. Each experiment conducted 100 different runs, and for each run accuracy and time consumption including training and test were outputted. The area under the curve (AUC) of the receiver of operating characteristics curve (ROC) [**35**] was used to evaluate the accuracy of classification as well as precision, recall, and f1-score [**114**]. Precision, recall, and f1-score are outputted for each class and class-wise weighted averaged results were provided to take class imbalance into account. All experiments were conducted on Python [Version 3.7.6].

2.3.4.3. *Training details with parameter tuning.* There were hyperparameters for each ML algorithm that could affect its accuracy and performance. Hyperparameters were tuned for optimal performance. The training details for each ML algorithm experiment were as follows: 1) RF experiments. 100 trees were used, and a max of depth was not assigned, which means nodes were expanded until all leaves were pure or all leaves contained less than the minimum sample split samples (fixed as 2 in our experiments). 2) SVM experiments. Linear kernel, a main hyperparameter, was adopted and 0.0001 was used as a learning rate, and also max iteration was not determined. 3) KNN experiments. k=5 was selected after comparison with other integers. 4) MLP experiments.

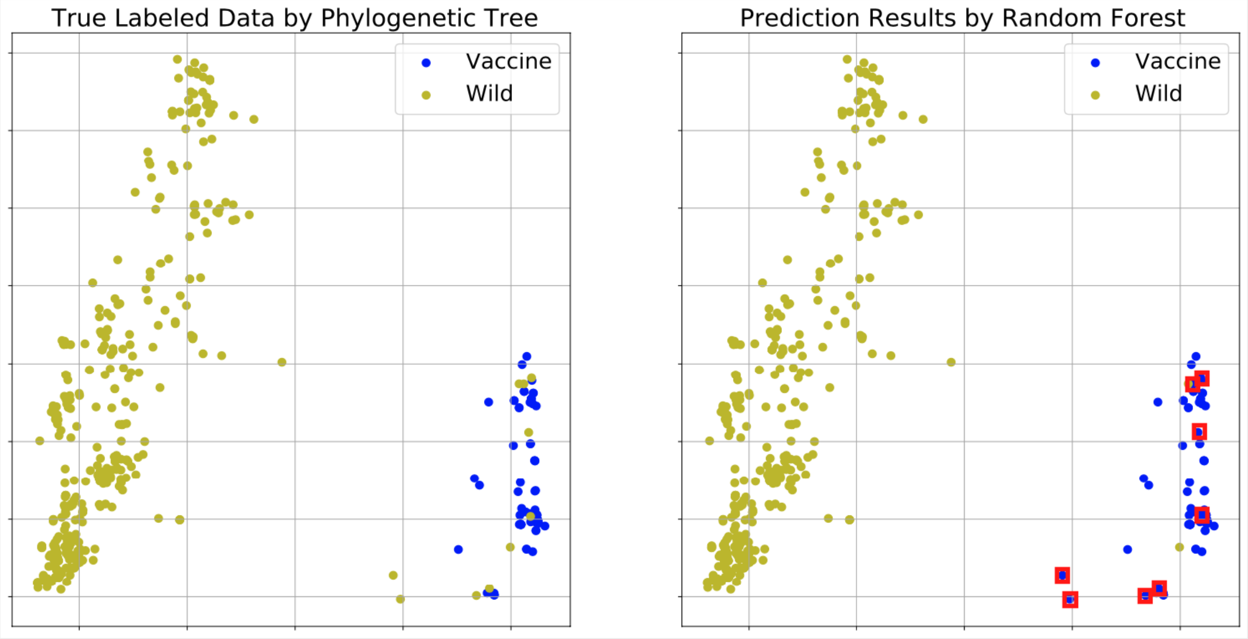


Figure 3: Principal component analysis (PCA) visualizations of random forest prediction results. Yellow and blue dots represent wild and vaccine type PRRSV strains, respectively. Two different types of PRRSV strains belonged to distinct clusters with a huge margin in between.

One hidden layer with 128 nodes followed by the ReLU activation function was used, and the softmax function was used for the output layer.

2.4. Results

2.4.1. Phylogenetic analysis for labeling and RF classification and detection of key amino acid positions.

2.4.1. Phylogenetic analysis for labeling and RF classification and detection of key amino acid positions. The phylogenetic analysis of 1931 field PRRSV strains using the ORF5 gene showed two distinct clades involving 438 L5 clade (22.6%) and 1498 L1 clade (77.4%) (Figure 1). Field PRRSV strains in L1 clade were further classified into one of three Sublineages (L1A clade: Sublineage 1.5, L1B clade: Sublineage 1.6, and L1C clade: Sublineages 1.7, 1.8 and 1.9). PCA visualization of the classification presented the clear margin between clusters of L1 and L5 clades (Figure 3). However, we observed contiguous margins for the classification among L1A, L1B, and L1C clades. Importance scores of RF classification in each amino acid position were outputted by RF. RF found that highly right-skewed distribution of importance score in amino acid positions (Figure 4) and four amino acid positions showed notably higher importance scores than other positions (> 0.06)

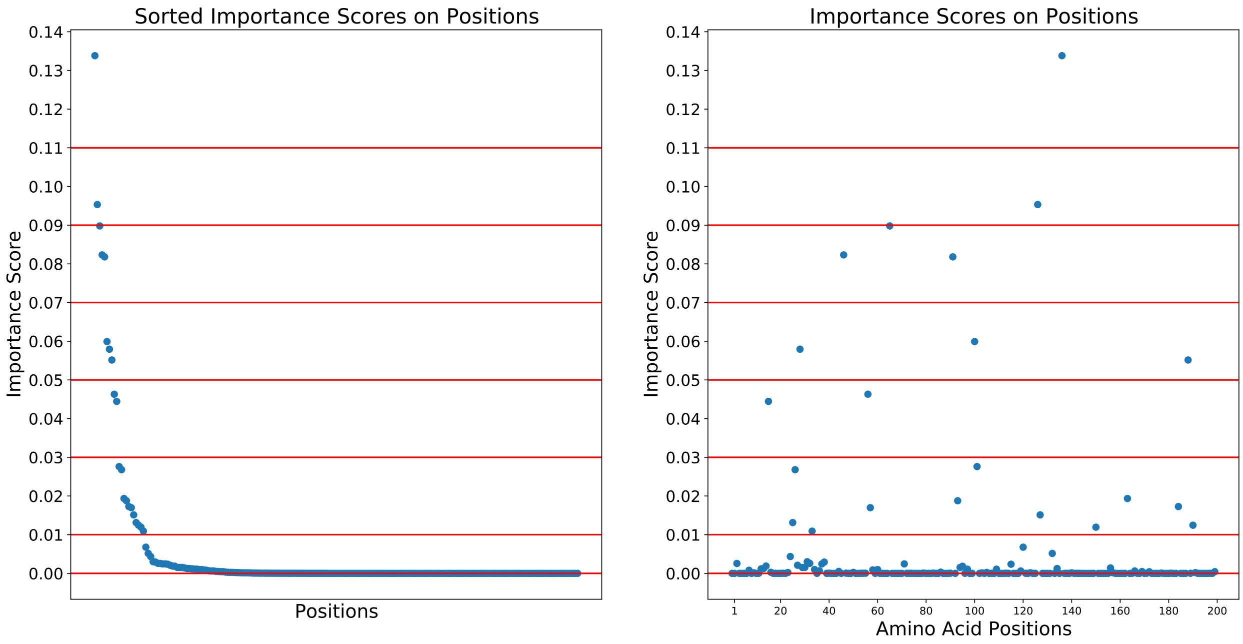


Figure 4: Distribution of importance scores for random forest differentiation of field PRRSV strains between the vaccine and wild types. (A) The importance scores in decreasing order for 200 amino acid positions with three cut-off thresholds; (B) Importance scores by the amino acid sequences.

Amino acid position	Importance Score	Amino acid in L5 clade (%) (n=438)	Amino acid in L1A clade (%) (n=1225)	Amino acid in L1B clade (%) (n=69)	Amino acids in L1C clade (%) (n=199)

26	0.145	A(98.4)
		V(0.9)
		T(0.7)
			V(98.3)
			I(1.6)
			A(0.2)
				V(94.2)
				A(5.8)
					A(96.5)
					D(0.5)
					V(3.0)

170	0.071	E(99.5)
		G(0.5)
			E(98.1)
			G(1.6)
			K(0.3)
				N(100)	G(100)

137	0.070	A(99.8) X(0.2)*	S(100)	S(100)	S(100)

191	0.063	R(99.5)
		Q(0.2)
		X(0.2)*
			K(99.8)
			X(0.2)*
				K(100)	R(64.8)
					K(34.7)
					S(0.5)

X^* is sequencing errors.

Table 2.1: Top 4 key amino acid positions in open reading frame 5 genes with the highest random forest importance scores for the 1931 field PRRS strain classification [Importance score > 0.06].

[26th, 170th, 137th, and 191st] (Table 2.1). The 26th position showed significant heterogeneity of amino acid sequences between L5 (A: 98.4%) and L1A (V: 98.3%) clades. The amino acid sequence

ML # of Pos.	RF	SVM	KNN	MLP
200	0.99/0.99/0.99	0.99/0.99/0.99	0.99/0.99/0.99	0.99/0.99/0.99
4	0.99/0.99/0.99	0.99/0.99/0.99	0.99/0.99/0.99	0.99/0.99/0.99
3	0.99/0.99/0.99	0.99/0.99/0.99	0.99/0.99/0.99	0.99/0.99/0.99
2	0.99/0.99/0.99	0.99/0.99/0.99	0.98/0.98/0.98	0.99/0.98/0.98
1	0.75/0/85/0.79	0.75/0/85/0.79	0.73/0/83/0.77	0.73/0/83/0.77

Table 2.2: The class-wise averaged approximated precision/recall/f1-score values for the corresponding four machine learning (ML) algorithms by five experiments.

of L5 (E: 99.5%) and L1A (E: 98.1%) clades in the 170th position was also heterogenous compared to L1B (N: 100%) and L1C (G: 100%) clades. The 137th amino acid position was a key site to classify between L5 (A: 99.8%) and three L1 (S: 100%) clades (Table 2.1).

RFLP analysis classified our 1931 field PRRSV strains into 43 types [Supplement 1]. The RFLP type classified all 1931 PRRSV strains into either L5 (7 types) or L1 (36 types) clades correctly. Almost all strains in L5 clades were classified into 2-5-2 RFLP type (93.6%, 410/438). PRRSV strains in one of three L1 clades were classified into either 1-8-4 (59.3%, 888/1498), 1-7-4 (19.9% 298/1498), or 1-4-4 (6.8% 102/1498) RFLP type. However, for the classification of three L1 clades, field PRRSV strains in 9 RFLP types (1-3-2, 1-3-4, 1-4-3, 1-4-4, 1-7-4, 1-8-3, 1-8-4, 1-12-4, & 1-16-4) belonged to two L1 clades at the same time (L1A L1C clades or L1B & L1C clades) [Supplement 1].

2.4.2. The Accuracy and time consumption of classification using four ML algorithms by the number of amino acid sites. We performed five ML experiments including 1) the data fully utilizing 200 amino acid positions, and 2) the four data sequentially adding amino acids from the position with the highest RF score (26th) to the fourth highest RF score (191st) [Table 2.1]. All five ML experiments showed high accuracy for the classification of field PRRSV strains except one experiment using only the 26th amino acid position (Figure 5). In the four experiments with 2 or more than 2 amino acid positions (2/3/4/200), all four ML approaches showed approximately 100% accuracy in terms of AUC, precision, recall, and f1-score (Table 2.2). However, in the experiment using one amino acid sequence with the highest importance score, the 26th amino acid position, the accuracy decreased drastically to approximately 80% in all ML four methods. KNN showed a high variability in accuracy but the other three ML approaches had relatively low variability. In the subsequent experiment adding one additional amino acid sequence with the second highest

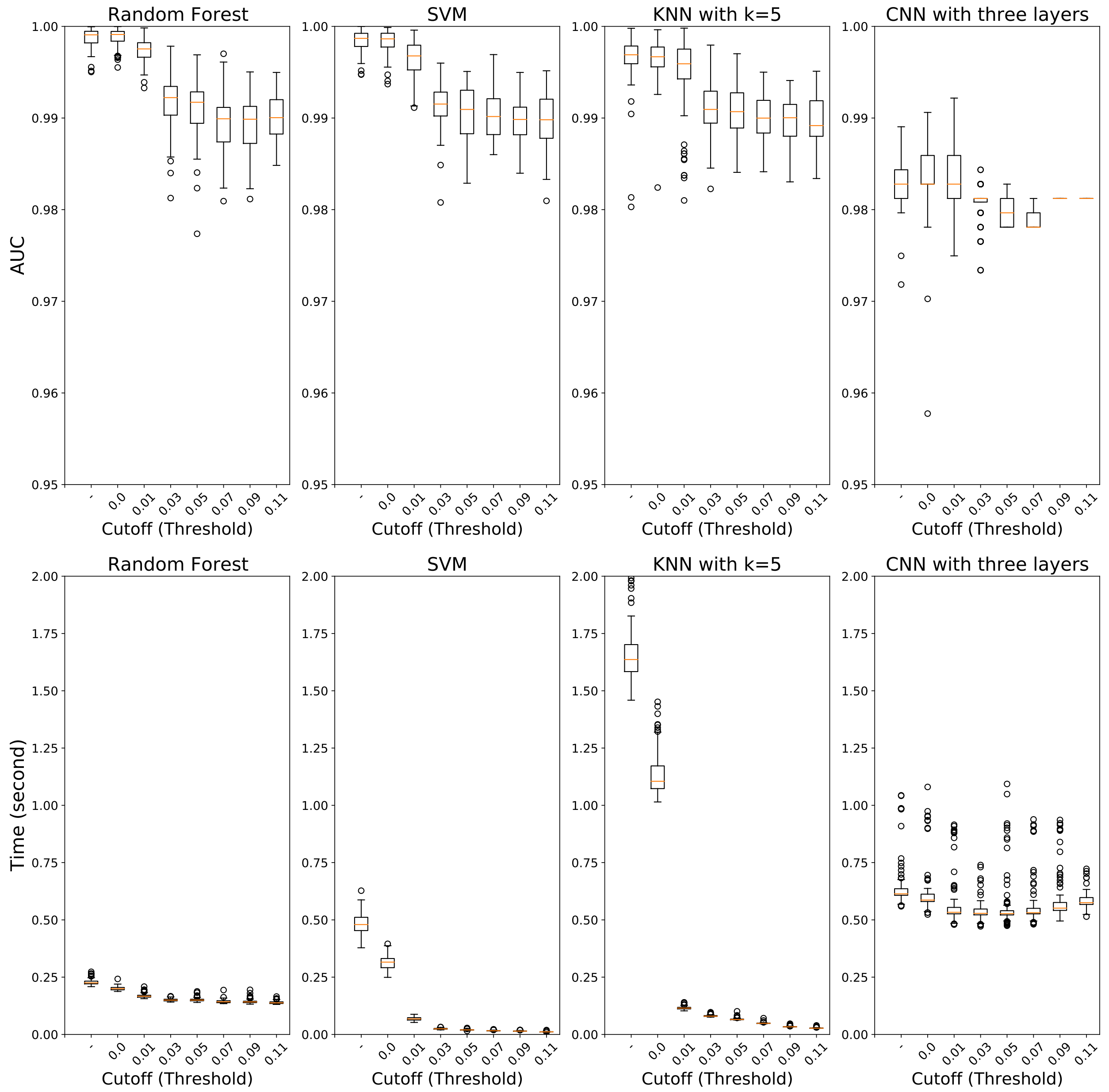


Figure 5: Experimental results of four machine learning algorithms using boxplot with two different ratio of training and test data size. For each of (A) and (B), Top: area under the curve (AUC) values. Bottom: Time consumption (seconds). Orange lines are mean values over 100 runs.

importance score, 170th position, all ML four methods showed very high accuracy of classification, as the experiment using all 200 amino acid positions did. Four ML algorithms classified field PRRSV strains in very short time consumption ($<$ 2.5 seconds). RF showed consistently short time consumption even with the changes in the number of amino acid sequences used. However, SVM and KNN required higher time consumption when they worked with all 200 amino acid sequences. MLP showed high variability in the time consumption with consistency as the number of used positions changed (Figure 5).

2.5. Discussion

The present study demonstrated that ML algorithms enabled to classify US field PRRSV strains into four clades accurately using five amino acid scores transformed from the ORF5 gene sequences with short time consumption. Furthermore, one of four ML algorithms, specifically RF, was used to detect key amino acid positions potentially associated with biological characteristics of PRRSV strains.

In the present study, all four ML approaches accurately classified four clades even using small genetic information. Although each field PRRSV strain involved high-dimensional genome data including 1000 features (5 scores x 200 amino acids), PCA visualization depicted that the genetic difference of field PRRSV strains between L5 and three L1 clades were distinctly distinguished. However, the genetic contiguity among L1A, L1B and L1C clades posed a challenge for the classification of field PRRSV strains. In the classification using one amino acid sequence in the 26th position with the highest RF importance score, all ML approaches showed fairly high AUC value (> 0.79), and, RF, SVM, and MLP classified field PRRSV strains stably compared to KNN. Considering the significant heterogeneity of amino acid composition, the 26th position played a key role as a classifier between L5 and three L1A clades which constituted 83.9% of our PRRSV samples. Interestingly, after we additionally included one amino acid sequence in the 170th position, all four ML algorithms showed stable and very high accuracy for the classification (AUC > 0.99). Although the 170th amino acid position showed homogeneity between L5 and L1A, this position showed significant heterogeneity among L1A, L1B, and L1C clades. Consequently, the combination of 26th and 170th amino acid sequences provided sufficient information for all ML approaches to identify the best-fit decision boundaries of classification among four clades. In the perspective of the accuracy of classification, any of the four ML approaches outperformed the RFLP typing. Specifically, considering the very high stability of classification accuracy using all 200 amino acid sequences, RF might be a prioritized option to handle the high-dimensional genome data because RF, an ensemble ML algorithm, builds sufficiently a large number of decision trees and minimizes overfitting.

ML algorithms also have a great benefit in the time consumption compared to the phylogeny estimation. Generally, the classification using the phylogenetic analysis of infectious pathogen based on a large number of genome sequences requires high computational power and subsequent time consumption because the phylogeny estimation searches the unrooted phylogeny with the highest likelihood among possible unrooted phylogenies and the number of unrooted phylogenies gets exponentially increased by the number of sequences. However, all four ML approaches require a very short amount of time for model training and classification of test data even with a very large number of PRRSV sequences ($<$ 2.5 seconds). Specifically, RF and MLP showed high consistency in time consumption regardless of the number of features. Even with the small number of features, RF requires a fair amount of time to generate a large number of decision trees for stable model training. MLP also needs many computational steps to catch the underlying characteristics of the data. However, considering the consistency of short and constant time consumption, RF and MLP could be well-adapted for the classification of large genome data with high complexity rather than SVM and KNN.

The RF algorithm was used to detect key amino acid substitutions potentially associated with the biochemical characteristics of PRRSV. The 26th and 137th amino acid positions had high importance score with significant heterogeneity between L5 and three L1 clades. The 26th position of the ORF5 gene showed the highest importance score and was located in one of two cleavage sites in the decoying epitope of the GP5 protein [**111**]. A previous study found that the amino acid substitution in the 26th position influenced on the host antibody response against PRRSV infection and characterized the infectivity of a PRRSV strain [**84**]. In the 137th position with the third highest importance score, all L5 clade strains had Alanine and three L1 clade strains substituted to Serine. Alaine in the 137th position of the ORF5 gene is generally monitored as a marker of Ingelvac PRRS Type 2 MLV in the L5 (Boehringer Ingelheim Vetmedica Inc., St. Joseph, Missouri, USA) because the substitution of Alaine to Serine in the 137th position of ORF5 gene considerably reduced the susceptibility of viral neutralization against VR2332 anti-serum, the reference strain of Ingelvac MLV6,23.

Although all four ML approaches showed very high accuracies in the classification of field PRRSV strains, strong genetic homogeneity within clades compared to heterogeneity among clades possibly inflated the accuracy of this study. The present study observed significant genetic heterogeneity among four clades of PRRSV strains, especially in the key 4 amino acid positions of ORF5 gene. Consequently, all four ML approaches led to nearly perfect accuracy for the classification even including two key amino acid sequences. It implies that our ML approaches potentially showed lower accuracy and longer time consumption in the multi-class classification with high-complexity genome data. In future research, we will explore ML approaches for more complex classification using larger genetic information such as the prediction of multiple phenotypic and antigenic characteristics classification to evaluate the accuracy and time consumption of ML approaches and the detection of key substitutions related to unique biological characteristics by each clade of field PRRSV strains.

In the modern livestock industry, genome sequencing enables to obtain of high-quality and large genetic information on infectious pathogens and is widely applied to the genome-based diagnostics of infectious pathogens. This study exemplified the use of high-quality genetic information for the classification of phenotypic characteristics of infectious pathogens. Once ML algorithms were sufficiently trained for classification, all ML algorithms accurately classified the genetic characteristics in a very short time and detected key amino acid sites, specifically for the rapid vaccine lineage selection based on genetic relatedness at a pig farm level (e.g. Prevacent® for lineage 1 and Ingelvac for lineage 5). We believe that our ML approaches using the amino acid scores for the classification of field PRRSV strains can be applied as a powerful tool in the digitalized surveillance system considering its very short time consumption and high accuracy. Furthermore, the use of ML approaches coupled with genetic information as we presented may inform decision-makers in the US pig industry to have a better understanding of PRRSV evolution and transmission dynamics and establish cost-effective control and preventive measures of PRRSV using MLV at farm or production system level.

2.6. Conclusion

This study proposed the use of ML algorithms for the classification of field PRRSV strains into four clades and the detection of the key amino acid substitutions in the ORF5 gene. Our

ML approaches showed very high accuracy and short time consumption compared to conventional approaches of PRRSV classification. We believe that our ML approaches based on amino acid score could be a powerful alternative to handle large genome datasets in real time or semi-real time to classify field PRRSV strains as well as other infectious pathogens and support decision-making or design more timely surveillance or intervention strategies.

2.7. Supplements

Supplement 1. The classification of 1931 field PRRSV strains into four clades (L1A, L1B, L1C and L5 clade) using the RFLP analysis of ORF5 gene.

RFLP	L1A (n=1225)	L1B (n=69)	L1C (n=199)	Total
1-1-1			1	1
1-1-2		30		30
1-10-2	1			1
1-10-4	9			9
1-12-4	3		4	7
1-16-2	1			1
1-16-4	6		1	7
1-18-2		2		2
1-18-3		1		1
1-19-4		3		3
1-2-2			2	2
1-2-4			11	11
1-21-4	5			5
1-24-2		2		2
1-26-2		28		28
1-26-4		2		2
1-3-1	1			1
1-3-2		1	1	2
1-3-3	1			1
1-3-4	8		1	9
1-30-4	1			1
1-33-3	3			3
1-33-4	1			1
1-4-1	6			6
1-4-3	1		6	7
1-4-4	35		67	102
1-45-4	1			1

RFLP	L5 (n=438)	L1A (n=1225)	L1C (n=199)	Total
1-5-2	10			10
1-6-2	1			1
1-6-4		3		3
1-7-2		15		15
1-7-4		297		297
1-8-1		1		1
1-8-2		10		10
1-8-3		27	3	30
1-8-4		788	100	888
1-8-8		1		1
1-45-4			1	1
2-1-2	9			9
2-5-1	3			3
2-5-2	410			410
2-5-4	2			2
2-6-2	3			3

CHAPTER 3

Predicting Antimicrobial Resistance of Bacterial Pathogens using Time Series Analysis

Published in *Frontiers in Microbiology* (May 2023).

Edited for this dissertation.

Joint work with:

Ruwini Rupasinghe, Beatriz Martínez-López

Department of Medicine and Epidemiology, Center for Animal Disease Modeling and Surveillance (CADMS), School of Veterinary Medicine, University of California, Davis, Davis, CA, United States

{pvmlee, rkrupasinghe, beamartinezlopez}@ucdavis.edu

Avisahi Halev

Department of Mathematics, University of California, Davis, Davis, CA, United States

ahalev@ucdavis.edu

Chao Huang, Shahbaz Rezaei, Xin Liu

Department of Computer Science, University of California, Davis, Davis, CA, United States

{srezaei, xinliu}@ucdavis.edu

Maria J. Clavijo

Department of Veterinary Diagnostic & Production Animal Medicine (VDPAM), Iowa State University, Ames, Iowa, United States

mclavijo@iastate.edu

Rebecca C. Robbins

R.C. Robbins Swine Consulting Services, PLLC, Amarillo, TX, United States

rebecca.robbins@genusplc.com

3.1. Abstract

Antimicrobial resistance (AMR) is arguably one of the major health and economic challenges in our society. A key aspect of tackling AMR is rapid and accurate detection of the emergence and spread of AMR in food animal production, which requires routine AMR surveillance. However, AMR detection can be expensive and time-consuming considering the growth rate of the bacteria and the most commonly used analytical procedures, such as Minimum Inhibitory Concentration (MIC) testing. To mitigate this issue, we utilized machine learning to predict the future AMR burden of bacterial pathogens. We collected pathogen and antimicrobial data from >600 farms in the United States from 2010 to 2021 to generate AMR time series data. Our prediction focused on five bacterial pathogens (*Escherichia coli, Streptococcus suis, Salmonella sp., Pasteurella multocida,* and *Bordetella bronchiseptica*). We found that Seasonal Auto-Regressive Integrated Moving Average (SARIMA) outperformed five baselines, including Auto-Regressive Moving Average (ARMA) and Auto-Regressive Integrated Moving Average (ARIMA). We hope this work provides valuable tools to predict the AMR burden not only of the pathogens assessed in this study but also of other bacterial pathogens.

3.2. Introduction

The discovery of antimicrobials is one of the best advances in therapeutic medicine in humans and animals. Over time microbes have evolved and developed resistance mechanisms against these antimicrobial compounds. Increasing resistance to the available antimicrobials and stagnation of developing novel antimicrobials limit treatment options for patients with infectious diseases. Therefore, the emergence, dissemination, and persistence of microbes that are resistant to existing antimicrobials pose an enormous threat to public and animal health. Antimicrobials are extensively used in the food animal industry to treat bacterial infections and promote health, welfare, and production. According to Food and Drug Administration (FDA), approximately 80% of all antibiotics sold in the United States in 2011 were sold for use in animal husbandry, and around 70% of them belonged to the antibiotic classes used in human medicine (medically-important antibiotics) [**28**]. Pig farming is one of the leading sectors using antimicrobials. Thus, increased levels of AMR are anticipated in swine farms due to the selective pressure of these antimicrobials and can spread via pork, direct contact with pigs, or discharge of swine waste into the environment.

A key to preventing AMR emergence and spread is early and accurate detection of potential AMR, which promotes selecting appropriate antimicrobials and facilitating the prompt investigation of drug-resistant disease outbreaks. Routine monitoring and surveillance can enable exemplary stewardship by detecting AMR emergence, tracing AMR patterns, and effectively targeting antimicrobial interventions and mitigation strategies. Currently, antimicrobial susceptibility testing (AST) is the primary method for detecting AMR and selecting effective antimicrobials against bacterial pathogens, which involves culturing the bacteria in the presence of a panel of various antimicrobials. Effective antimicrobials can be determined by detecting Minimum Inhibitory Concentration (MIC), where antimicrobials with lower MIC values are considered more effective (susceptible) because less of the drug is needed to inhibit bacterial growth. However, these procedures can be expensive and time-consuming, depending on the growth rate of the bacteria and MIC testing procedures. Alternative methods, such as DNA sequencing technologies, are increasingly used to detect AMR at the molecular level, but they require robust bioinformatics tools to evaluate the genomic structure of the microbial resistomes. Thus, most clinical laboratories still depend primarily on conventional AST to conduct clinical therapy and observe AMR over time. Nevertheless, most farms may not have the resources (e.g., time and budget) to perform routine testing to detect AMR and quantify the AMR burden in field settings. Therefore, developing a tool to predict AMR burden based on available data, such as prior AMR information (susceptible/resistance) against common antimicrobials, could be very useful to better inform decision-making about antimicrobial use at the farm level, which consequently helps mitigate AMR.

Machine learning has been widely employed for studying AMR, highlighting its importance in predicting resistance levels mainly using features directly from genotypes [**72**, **81**, **82**, **90**, **118**]. However, there are situations where we do not obtain genomic data to predict AMR levels but only preserve historical phenotype information. Time series analysis is a great solution to relevant tasks for such situations. Time series has shown great performances in studying AMR [**2**,**33**,**42**,**44**,**64**,**109**] and sometimes their methods are limited to Auto-regressive Integrated Moving Average (ARIMA) [**12**] or it subcategory methods that cannot properly incorporate seasonal behavior of AMR levels. Among many time series approaches, the Seasonal Auto-regressive Integrated Moving Average (SARIMA) [**12**] has received significant attention because of its outstanding performance in time series forecasting. SARIMA shows its usefulness when some degrees of seasonality-periodic fluctuations occur repeatedly in the time series.

In this study, we used the SARIMA algorithm to predict the future burden of AMR (AMR proportions) of five bacterial pathogens (*Escherichia coli, Streptococcus suis, Salmonella sp., Pasteurella multocida,* and *Bordetella bronchiseptica*) prevalent in the studied swine farms using the prior resistance information. The data included the number of tested pathogens with confirmed resistance (based on MIC interpretations) to their corresponding antimicrobials. Instead of direct use of binary (susceptible and resistance) classified data, we generated integrated time series data, i.e., quarterly-based AMR proportions for each of the study pathogens. This approach enabled us to overcome the limitations of missing data over time. We also compared the performance of SARIMA to that of Auto-Regressive Moving Average (ARMA) [**122**], Auto-Regressive Integrated Moving Average (ARIMA) [**12**], and three other forecasting baseline methods: Naïve, Seasonal Naïve, and one-lagged prediction [**96**, **99**]. These three baselines were selected as benchmarks in our study because they are often used in forecasting tasks and are simple yet effective. We believe that predicting AMR proportions using time series models can provide valuable information to facilitate the selection of appropriate antimicrobials against pathogens and the prompt investigation of drug-resistance disease outbreaks.

3.3. Materials and Methods

In this section, we discuss the workflow, time series analysis methods, and experimental design. Workflow after data collection includes data processing (irregular binary data to quarterly time series data) and time series analysis (model parameter selection and model train/test) (Figure 1)

3.3.1. Data Collection. In this study, we used pathogen and antimicrobial information from >600 farms owned by two swine production systems in the United States. The samples were collected from pigs infected with one of five bacterial pathogens (*Escherichia coli, Streptococcus suis, Salmonella sp., Pasteurella multocida,* and *Bordetella bronchiseptica*) from 2010 to 2021 and tested for AMR against a panel of antimicrobials (Table 3.1). The resistance level of each pathogen

	Escherichia coli	Streptococcus suis	Salmonella sp.	Pasteurella multocida	B. bronchiseptica
Clindamycin	1.0/0.0	0.8/0.18	1.0/0.0	1.0/0.01	1.0/0.0
Tiamulin	0.99/0.03	0.16/0.1	1.0/0.01	0.58/0.29	1.0/0.01
Tylosin	-	-	-	0.98/0.06	-
Ampicillin	0.71/0.22	0.03/0.06	0.58/0.27	0.03/0.07	1.0/0.02
Gentamicin	0.32/0.16	-	0.51/0.39	-	0.04/0.1
Oxytetracycline	0.88/0.13	0.93/0.1	-	0.23/0.26	0.03/0.1
Penicillin	1.0/0.0	0.18/0.13	1.0/0.0	0.19/0.28	1.0/0.0
Spectinomycin	0.9/0.22	-	-	-	-
Tilmicosin	0.99/0.03	0.73/0.21	1.0/0.0	0.21/0.31	-
Chlortetracycline	0.88/0.13	0.93/0.1	-	0.03/0.07	0.04/0.11
Sulphadimethoxine	-	0.61/0.22	-	-	0.97/0.08
Ceftiofur	-	0.04/0.06	-	0.0/0.02	-
Enrofloxacin	0.34/0.25	0.07/0.07	0.29/0.38	0.0/0.02	-
Florfenicol	0.84/0.09	0.03/0.07	0.9/0.14	0.02/0.07	0.83/0.17
Neomycin	0.34/0.25	0.73/0.17	0.57/0.39	0.07/0.14	-
TRIMETHSULFA	0.26/0.27	-	0.32/0.32	-	0.89/0.14
Tulathromycin	-	-	-	0.0/0.01	0.04/0.1

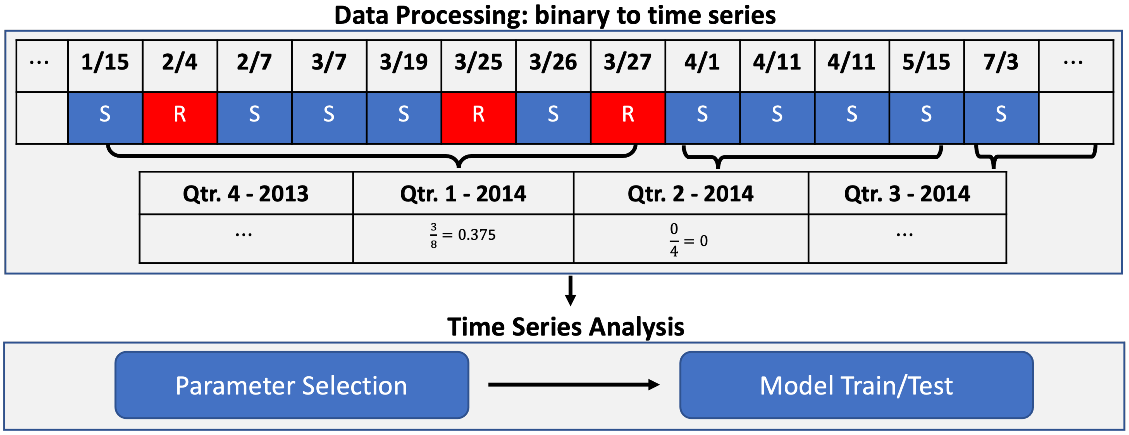
Table 3.1: Used antimicrobials and pathogens pairs for analysis. Provided number indicates the mean and standard deviation of the dataset (i.e., an indicator for averaged AMR proportions and how dynamically (uncertain) the time series is changing, respectively).

against antimicrobials was detected by determining MIC and classified into two groups: susceptible (S) and resistant (R), based on an interpretation report received from the American Association of Veterinary Laboratory Diagnosticians (AAVLD) accredited laboratory in the United States.

3.3.2. Data Processing for Time Series Analysis. For each pathogen, different groups of antimicrobials were employed for experiments (see Table 3.1). One challenge is that there were missing data points between certain time periods. To tackle this, we constructed quarterly time series dataset by integrating the data points every quarter. We converted our data points to the quarterly basis dataset and define $Res(Pathogen, Antimicrobial)$ the resistance time series for each pathogen and antimicrobial as below.

$$Res(Pathogen, Antimicrobial) = (r_1, r_2, \cdots, r_n), \tag{3.1}$$

where $r_i = Proportion(R) = \frac{\#\text{ of } R}{\#\text{ of } (R+S)}$ over the ith quarter (R and S stand for resistant and susceptible, respectively). Figure 1 shows how we processed our dataset. Figure 2 shows examples of quarterly-based time series constructed for pathogens and antimicrobials and all of the time series



Data Processing: binary to time series

…	1/15	2/4	2/7	3/7	3/19	3/25	3/26	3/27	4/1	4/11	4/11	5/15	7/3	…
	S	R	S	S	S	R	S	R	S	S	S	S	S	

Qtr. 4 - 2013	Qtr. 1 - 2014	Qtr. 2 - 2014	Qtr. 3 - 2014
…	$\frac{3}{8} = 0.375$	$\frac{0}{4} = 0$	…

Time Series Analysis

Figure 1: Workflow Chart. Data processing example (from irregular binary data to quarterly based time series: $Res(Pathogen, Antimicrobial)$) and time series analysis.

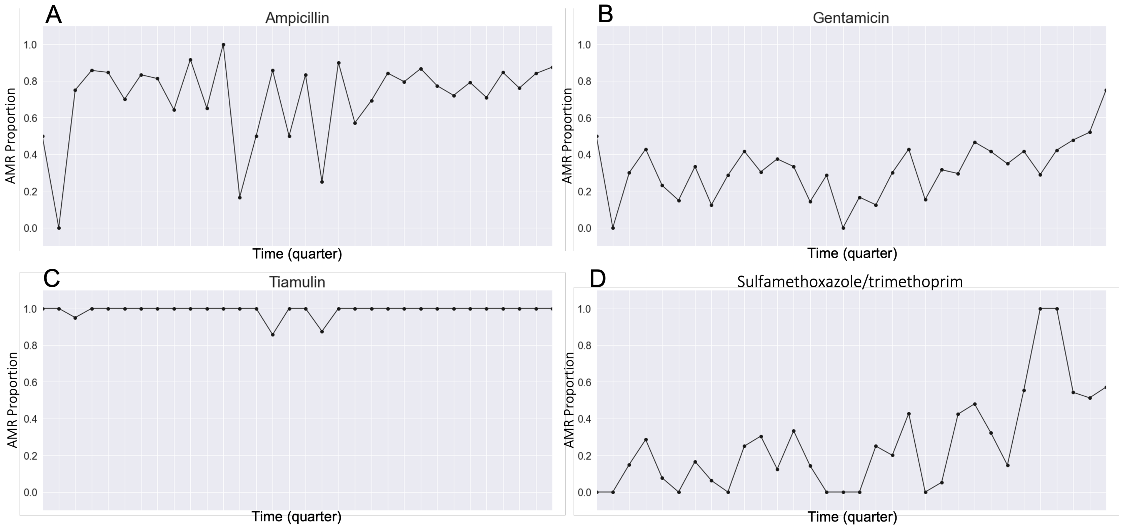


Figure 2: Examples of our processed quarterly-based AMR proportions time series. **(A)** *Res(Escherichia coli, Ampicillin)*, **(B)** *Res(Escherichia coli, Gentamicin)*, **(C)** *Res(Escherichia coli, Tiamulin)* and **(D)** *Res(Escherichia coli, Sulfamethoxazole/trimethoprim)*

examples are presented as solid lines in Supplementary Figure 1 - Figure 5. With the constructed data, we focused on predicting AMR proportions in times series for each $Res(Pathogen, Antimicrobial)$ in our data.

We also output the mean and the standard deviations of $Res(Pathogen, Antimicrobial)$ which can be an indicator for the averaged AMR proportions and dynamics of each time series. (Table 3.1). We also observed different degrees of fluctuation in the processed dataset. For example,

$Res(Escherichia\ coli, Ampicillin)$ changes more dynamically than $Res(Escherichia\ coli, Tiamulin)$ (Figure 2 and Table 3.1)

3.3.3. AR(I)MA, SARIMA, and Three Baselines.

3.3.3.1. *ARMA and ARIMA.* *ARMA* model consists of two parts: autoregressive (AR) and moving average (MA) part [**122**]. The model is usually referred to as $ARMA(p, q)$ where p and q are the order of the AR and MA part, respectively [**115**]. AR part takes previous observations as inputs to predict future values. MA part uses previous errors between predicted and observed as predictors for future values. *ARIMA* model consists of three parts: AR, MA, and the integrated (I) part [**12**]. The model is usually referred to as $ARIMA(p, d, q)$ where p and q are the same as for the ARMA model, and d is the degree of differencing. The integrated part refers to the differencing of observations to allow time series to become stationary.

3.3.3.2. *SARIMA.* *SARIMA* model [**12**], as an advanced method of ARIMA with a seasonal component, overcomes the limitation that ARIMA cannot tackle data with periodic behavior properly. In this study, we employed SARIMA to predict AMR proportions for bacterial pathogens considering AMR proportions vary over time with a potential seasonality.

A typical SARIMA model has seven parameters, referred as $SARIMA(p, d, q)(P, D, Q)_S$ where (p, q) and (P, Q) are the order of the non-seasonal and seasonal (autoregressive, moving) models, respectively, d and D are the numbers of non-seasonal and seasonal differences, respectively, and S is the periodic seasonality term. Choosing appropriate parameters is a key process for optimal SARIMA performance. Autocorrelation function and partial autocorrelation function to this end. To be precise, we first determine non-seasonality components (p, d, q) and then we find proper seasonal parameters $(P, D, Q)_S$ using the autocorrelation function and partial autocorrelation function. Time series datasets often have trends in time series and changes in the statistical structure of the series, which means non-stationarity. To find non-seasonality parameters, trend, and seasonality in time series should be removed using differencing techniques. After the removal of trend and seasonality, the autocorrelation function and partial autocorrelation function help determine non-seasonal parameters. Additionally, we also check the p-value between time series data and its lagged time series, and the number of lags with the lowest p-value determines seasonality parameter S for the SARIMA model. However, these steps do not always guarantee finding a specific set of parameters for the optimal

SARIMA model. In many cases, parameter exploration using grid search is required, which means that we set some possible candidates for parameters and check the SARIMA model performance to find sets of parameters with the best performance.

3.3.3.3. *Three baselines.* *Naïve* method is the simplest time series forecasting method where all remaining forecast is set equal to the observation made in the last timestamp as below.

$$F_{T+t} = Y_T \text{ for } t > 0, \tag{3.2}$$

where F and Y are forecastings and observed times series, respectively. T and $T + t$ are the timestamps of the last observation and the forecast time, respectively.

Seasonal Naïve method is an extension of the Naïve method with a seasonality. It predicts the forecasts based on the same timestamp in the previous cycle as below.

$$F_{T+t} = Y_{T+t-s(k-1)} \text{ for } t > 0, \tag{3.3}$$

where s is seasonality and k is completed cycles.

One-lagged prediction methods rely on the most recent acquired data [**99**]. One-lagged prediction utilizes the data from the previous timestamp to forecast the current timestamp as shown below.

$$F_{T+1} = Y_T \text{ for } t > 0. \tag{3.4}$$

3.3.4. Experiments.

3.3.4.1. *Parameter selection for SARIMA..* For accurate AMR time series prediction, it is crucial to find appropriate SARIMA parameters [**68**]. We selected *Escherichia coli* and *Neomycin* because $Res(Escherichia\ coli, Neomycin)$ provides the largest number of data points to work with, and it shows visible seasonality. We have seven parameters to determine: (p, d, q), (P, D, Q), and S. After using the differencing method to find parameter d and to remove trend component in $Res(Pathogen, Antimicrobial)$, autocorrelation function, partial autocorrelation function, p-value analysis, and parameter exploration were attempted to assess SARIMA parameters. We choose optimal SARIMA parameters that predict $Res(Escherichia\ coli, Neomycin)$ with the lowest error.

3.3.4.2. *ARMA and ARIMA Parameter Selection.* Similar to SARIMA, we also employed parameter exploration to find the optimal parameters for ARMA and ARIMA. $Res(Escherichia\ coli, Neomycin)$

were utilized for this process. We conducted two experiments for ARMA and ARIMA independently because ARMA does not take parameter d into account while ARIMA considers it.

3.3.4.3. *Time series-based AMR proportions prediction.* We selected seven combinations of parameters from previous analysis on $Res(Escherichia\ coli, Neomycin)$, and applied the chosen seven combinations of parameters to other $Res(Pathogen, Antimicrobial)$ to predict the AMR proportions. Specifically, for each $Res(Pathogen, Antimicrobial)$, seven experiments with different parameter sets were conducted. Each experiment returned a rooted mean squared error as a performance measurement. We also used three baselines: Naïve, Seasonal Naïve (we set 4 as the seasonality period), and One-lagged prediction. All baselines also outputted the rooted mean squared error values for each $Res(Pathogen, Antimicrobial)$. All experiments were conducted in Python (Version 3.7.6).

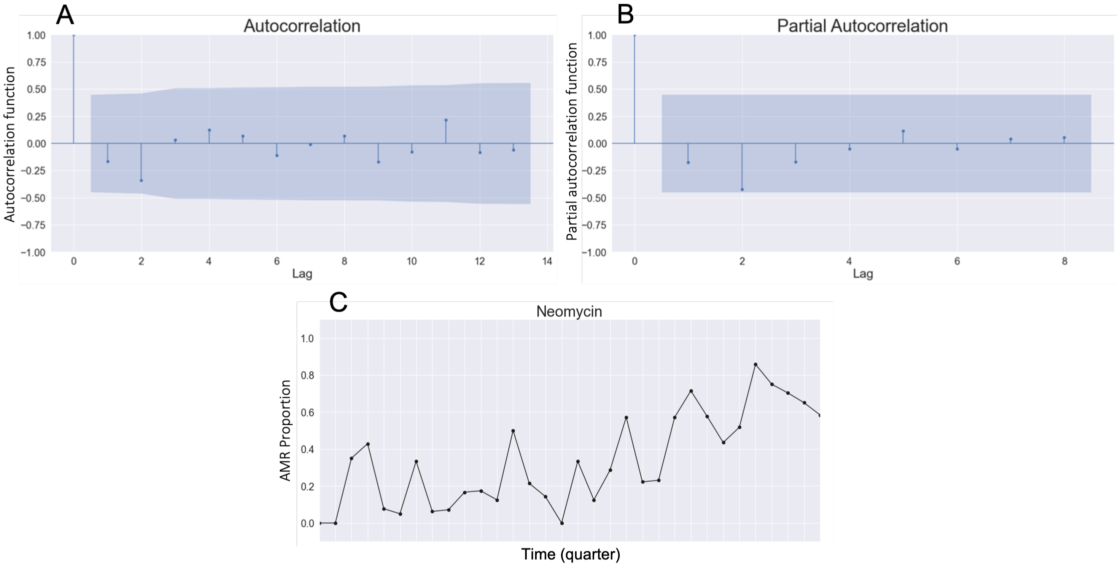


Figure 3: Autocorrelation function and partial autocorrelation function analysis for parameter selection. **(A)** autocorrelation function plot, **(B)** partial autocorrelation function plot and **(C)** AMR time series for *Escherichia coli* and Neomycin, i.e., *Res(Escherichia coli, Neomycin)*.

3.4. Results

3.4.1. Seven Selected Sets of SARIMA Parameters. As shown in Figure 3, the autocorrelation function and partial autocorrelation function provided information on choosing the right parameters for SARIMA. P-value analysis for the $Res(Escherichia\ coli, Neomycin)$ and its lagged

Methods	SARIMA							ARMA			ARIMA		
Parameters	p	d	q	P	D	Q	S	p	d	q	p	d	q
1	1	1	4	1	0	1	12	3	0	2	3	2	1
2	2	2	3	1	0	1	12	3	0	0	2	2	3
3	3	0	4	1	0	1	12	3	0	1	2	2	1
4	3	1	4	1	0	1	12	1	0	0	3	2	2
5	4	0	0	1	0	1	12	1	0	1	3	1	0
6	4	3	4	1	0	1	12	1	0	2	2	1	0
7	4	2	4	1	0	1	12	1	0	3	2	2	2

Table 3.2: Seven selected parameters of SARIMA, ARMA, and ARIMA used for overall AMR proportions prediction acquired from *Res(Escherichia coli, Neomycin)* analysis.

time series with different numbers of lags was also used to find the seasonal parameter S. From these, we can determine our parameter S = 12 but other parameters were not found properly from autocorrelation function and partial autocorrelation function analysis. There were no significant patterns of gradual decay or recurring cycles observed in either the autocorrelation or partial autocorrelation plots 3. Specifically, there is no data point with a lag value greater than zero that fell outside the confidence interval (blue shade area) in either plot 3, resulting in making it unable to estimate the appropriate parameters for a moving average (MA) or autoregressive (AR) models. From these analyses, the parameters of the time series model could not be satisfactorily determined without a parameter search.

In this regard, we explore the set of parameters that output the lowest error estimation measured by rooted mean squared error. In other words, we conducted trial and error for finding appropriate undetermined parameters remained. Our parameter exploration includes integers from 0 to 5 for three parameters p, d, q, and from 0 to 6 for the other three parameters P, D, Q, which we end up having $6^3 7^3$ combination to attempt. For each attempt with a combination of parameters, SARIMA predicted $Res(Escherichia\ coli, Neomycin)$, i.e., tried to predict the 10% of the last values in $Res(Escherichiacoli, Neomycin)$ after being trained with first 90% percent of the $Res(Escherichia\ coli, Neomycin)$ and, a prediction error for was reported. With outputted errors, we selected seven parameter combinations that return the lowest rooted mean squared error values because the next best one after these seven has a relatively big gap in the errors from the first seven parameters, and interestingly, our three parameters $(P, D, Q)_S$ are fixed as $(1, 0, 1)_{12}$ while seven different (p, d, q) are acquired from (Table 3.2).

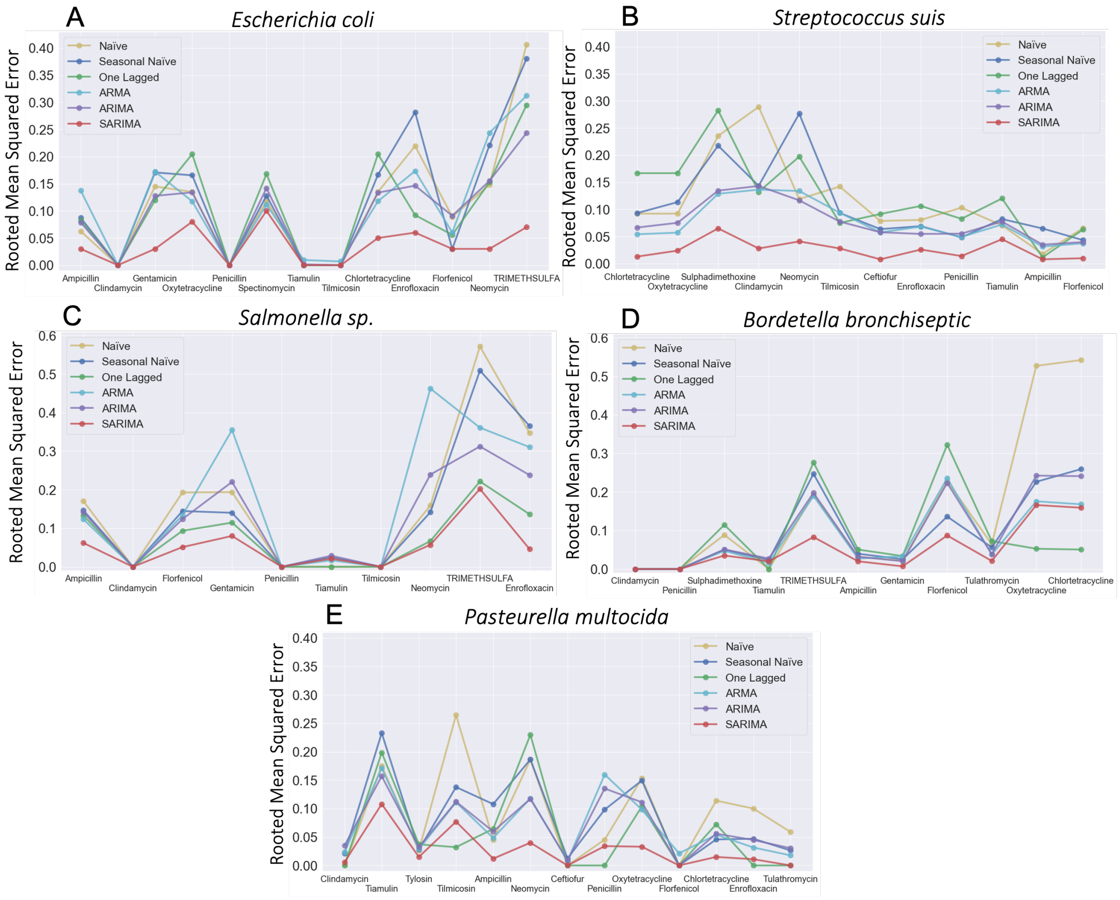


Figure 4: Rooted mean squared errors for five pathogens with corresponding antimicrobials. **(A)** *Escherichia coli*, **(B)** *Streptococcus suis* **(C)** *Salmonella sp.*, **(D)** *Bordetella bronchiseptica* and **(E)** *Pasteurella multocida*.

3.4.2. Seven parameter sets for ARMA and ARIMA. To find the parameter sets that predicted $Res(Escherichia\ coli, Neomycin)$ with the lowest errors, we explored integers from 0 to 5 for all parameters (p, q) and (p, d, q) for ARMA and ARIMA, respectively. Each experiment requires 6^3 and 6^2 iterations to search independently. In the end, seven sets of (p, d, q) and (p, q) that outputted the lowest rooted mean squared error was selected (Table 3.2).

3.4.3. Error estimation for AMR proportions Prediction. For each *Res(Pathogen, Antimicrobial)* time series prediction using SARIMA, the seven previously selected SARIMA parameter sets were applied. Each experiment outputted a rooted mean squared error value which represents how good the prediction is, i.e., the lower the rooted mean squared error value, the more accurate the method (Figure 4). The lowest error value was provided among seven errors from seven experiments of SARIMA for each *Res(Pathogen, Antimicrobial).* For each of ARMA and ARIMA, seven parameters were conducted, and the lowest rooted mean squared error values were outputted among seven different experiments. We observed that our SARIMA method showed lower rooted mean squared error values compared to ARMA, ARIMA, and the other three baselines in general. The rooted mean squared error gap between SARIMA and three baselines became bigger when the AMR proportions time series ($Res(Pathogen, Antimicrobial)$) has greater deviation values (equivalently, more dynamical). This is because higher deviation implies more fluctuation in AMR proportions time series that are harder to predict. For example, rooted mean squared error values were similar between SARIMA and three baselines for $Res(Escherichia\ coli, Tilmicosin)$ (standard deviation: 0.03) while the rooted mean squared error gap became bigger for $Res(Escherichia\ coli, Enrofloxacin)$ (standard deviation: 0.03)). (Figure 4 and Table 3.1).

3.5. Discussion

This study investigated the plausibility of executing data-driven forecasting of the future AMR burden using the available resistance data in >600 swine farms in the United States from 2010 to 2021. AMR burden was quantified quarterly by calculating the proportions of resistant strains of five crucial bacterial pathogens (*Escherichia coli, Streptococcus suis, Salmonella sp., Pasteurella multocida,* and *Bordetella bronchiseptica*) against their corresponding antimicrobials. The bacterial species assessed in this study were the most prevalent swine bacterial pathogens dispersed within the studied farms, significantly affecting their health, welfare, and productivity. These pathogens can cause various infections in pigs, including respiratory, gastrointestinal, and/or systemic infections, and antimicrobials are the primary mode of therapy and prevention of these infectious diseases [**98**]. Therefore, early and accurate detection of potential AMR of these pathogens is essential to determine the appropriate antimicrobials to use against and monitor for drug-resistant disease outbreaks. In this study, we used three machine learning-based time series analyses to predict the future AMR proportions in the studied farms and compared their performances to select the most efficient and accurate approach for future use. According to our findings, SARIMA predicted AMR proportions accurately and outperformed ARMA, ARIMA, and three baselines according to the rooted mean squared error value. However, parameter exploration remains a light limitation due to the potential computational burden because the key to prediction using SARIMA was to find appropriate parameters which cannot always be acquired from the general process using partial autocorrelation function.

According to this study, we observed distinct temporal trends in AMR proportions for the five pathogens against their corresponding antimicrobials during the study period (Supplementary Figure 1-5). For example, pathogens, such as *Escherichia coli* and *Salmonella sp.*, showed very high or increasing trends of AMR proportions against Enrofloxacin, Neomycin, Sulfamethoxazole/trimethoprim, and Clindamycin, etc., while *Streptococcus suis* exhibited low resistance to Ampicillin, Ceftiofur, Enrofloxacin, Florfenicol, and Tiamulin. Most of the studied antimicrobials are effective against *Pasteurella multocida*, whereas *Bordetella bronchiseptica* displayed higher resistance levels against most antimicrobials assessed in our study. Nevertheless, these quarterly based-AMR proportions showed frequent fluctuations in most pathogens against their corresponding antimicrobials throughout the study period (Supplementary Figure 1-5). Yet, our SARIMA models were able to correctly capture all these individual trends and predict the future AMR proportions with high accuracy. Specifically, our work demonstrated that SARIMA works well for dynamic time series, such as AMR proportion time series for the studied five pathogens even if it is difficult to fairly compare our results to those from other relevant studies as each system has its unique data samples and methods. Also, this method could be applied to predict other unexplored pathogens unless the available data is limited. In other words, this work can be generalized to AMR proportions time series for any pairs of pathogens and antimicrobials. Furthermore, our SARIMA model can also be applied to other time series analyses in the domain, such as swine mortality rate, etc.

Early detection of emerging AMR and future prediction of AMR burden and trends are vital to comprehend the extent of the threat and implement appropriate antimicrobial interventions and mitigation strategies. Numerous studies have explored various ML algorithms to study AMR using available phenotypic data [**2**, **33**, **42**, **44**, **64**, **109**] and genotypes [**72**, **81**, **82**, **90**, **118**]. Specifically, the recent advancements in affordable and rapid DNA sequencing technologies (e.g., whole genome sequencing) combined with ML approaches have drastically transformed AMR surveillance and prediction prospects. Forecasting pathogens that have AMR based on genomics data has shown promising for the real-time detection of AMR determinants. However, this process requires robust bioinformatics tools and advanced analytical skillsets to assess the microbial genomic structure and the resistomes, and these limitations still preclude cost-effective, user-friendly, and rapid antimicrobial resistance surveillance. Besides, phenotyping approaches provide direct visual evidence of interaction between a bacterial strain and an antimicrobial. Thus, most clinical laboratories, to date, rely mainly on traditional AST to guide clinical therapy and monitor AMR over time. Therefore, the SARIMA model we proposed in our study will be an efficient and practical alternative to predict AMR burden, especially for situations where we do not have genomic data but only have historical phenotype information.

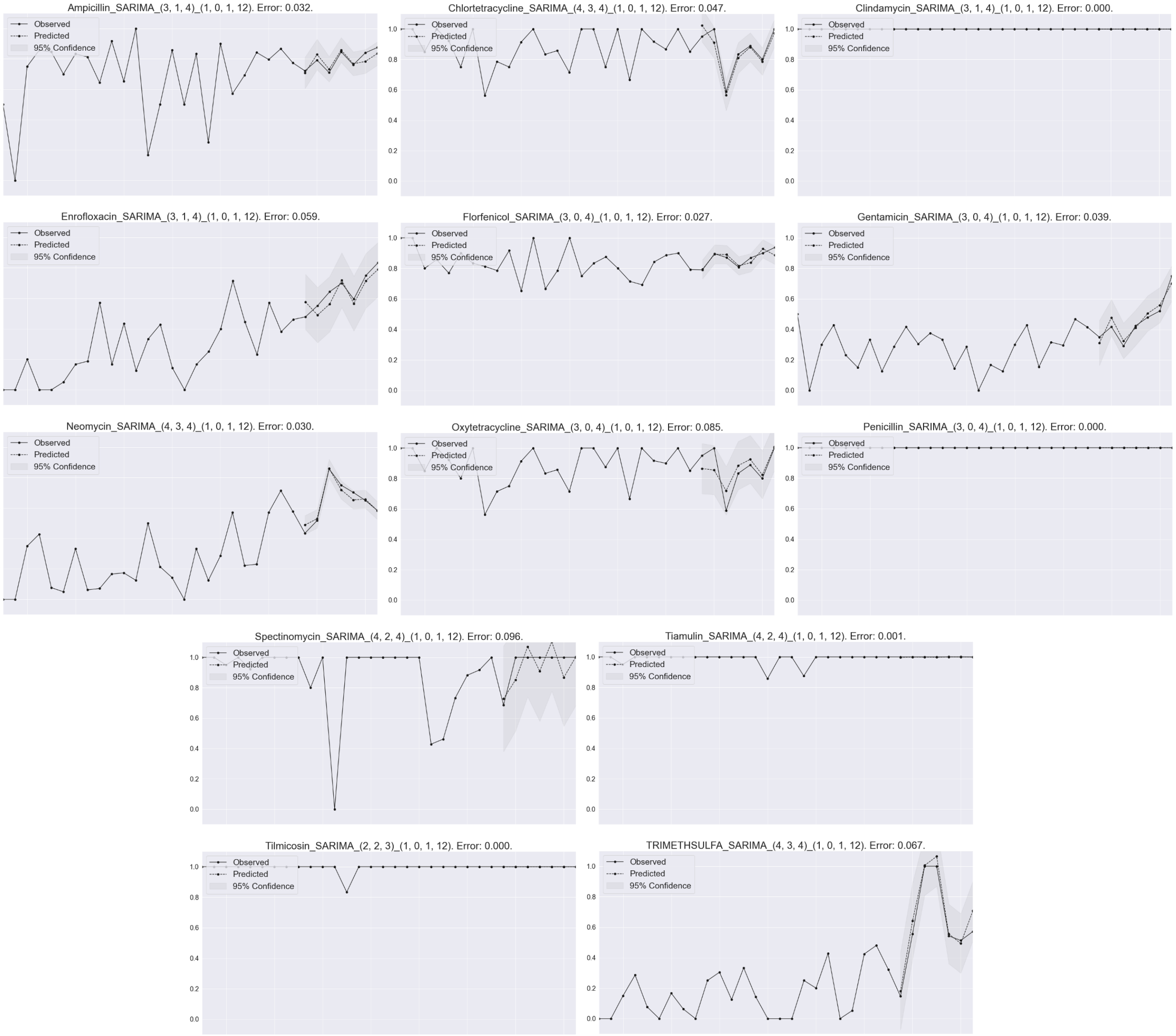
There are a few limitations to our study. The AMR data used for prediction was comprised of data from multiple swine farms within the United States. Although these farms were managed under two major swine production systems, individual farms can have different management practices, biosecurity measures, treatment protocols, etc. Previous studies disclosed various factors, such as transportation, farm management, housing conditions, metals consumption, feeding strategies, and antimicrobial usage that can affect the spread of antimicrobial-resistant bacteria and the AMR levels in a farm [**21**, **67**, **74**, **75**]. However, we did not incorporate these factors in our study. Thus, the future AMR burden (proportions) can vary from the predicted levels due to the variations in these farm factors. Since the AMR predictions were made using a limited number of swine farms in the United States, Therefore, we cannot generalize our findings to the entire swine population in the United States. Yet, our results depict the potential of using time series analysis to predict AMR levels within a farm or geographical region. In this study, we transformed the AMR data into a binary variable (susceptible/resistance) using breakpoints acquired from the interpretation report from AAVLD-accredited laboratories in the United States. Some of these breakpoints were extrapolated from other species (e.g., humans) if swine-specific breakpoints were not available for a pathogen-antimicrobial combination [**66**, **119**]. Breakpoint MICs depend on the clinical pharmacology of antimicrobials and are generally specific for bacterial-antimicrobial-host-disease-dosing regimen combinations [**66**, **119**], thus, different testing laboratories may use different standards for resistance classifications, which may cause misclassifications of pathogens. Nevertheless, predicting AMR burden directly from MIC values will minimize these misclassifications or classification errors. Hence, future studies are suggested to perform time series analysis based on the raw MIC data.

3.6. Conclusion

This study proposed to use time series methods for the prediction of future AMR burden by constructing the quarterly-based AMR proportions times series. The SARIMA approach showed low errors in terms of rooted mean squared error compared to ARMA, ARIMA, and three other forecasting baselines, and it worked even for highly dynamic time series. We believe that our time series prediction can help to advise using appropriate antimicrobials and reduce the risk related to AMR events by predicting anticipation of AMR occurrences in farms or geographical regions. Further, our study may also contribute to the analysis of similar problems and scenarios.

3.7. Supplements

Supplement 1. Prediction results on *E. Coli.* time series data samples based on the SARIMA (e.g. of 9 antibiotics). Solid and dotted lines are observed and predicted values, respectively, and grey area is 95% prediction intervals.



Ampicillin_SARIMA_(3, 1, 4)_(1, 0, 1, 12). Error: 0.032.

Chlortetracycline_SARIMA_(4, 3, 4)_(1, 0, 1, 12). Error: 0.047.

Clindamycin_SARIMA_(3, 1, 4)_(1, 0, 1, 12). Error: 0.000.

Enrofloxacin_SARIMA_(3, 1, 4)_(1, 0, 1, 12). Error: 0.059.

Florfenicol_SARIMA_(3, 0, 4)_(1, 0, 1, 12). Error: 0.027.

Gentamicin_SARIMA_(3, 0, 4)_(1, 0, 1, 12). Error: 0.039.

Neomycin_SARIMA_(4, 3, 4)_(1, 0, 1, 12). Error: 0.030.

Oxytetracycline_SARIMA_(3, 0, 4)_(1, 0, 1, 12). Error: 0.085.

Penicillin_SARIMA_(3, 0, 4)_(1, 0, 1, 12). Error: 0.000.

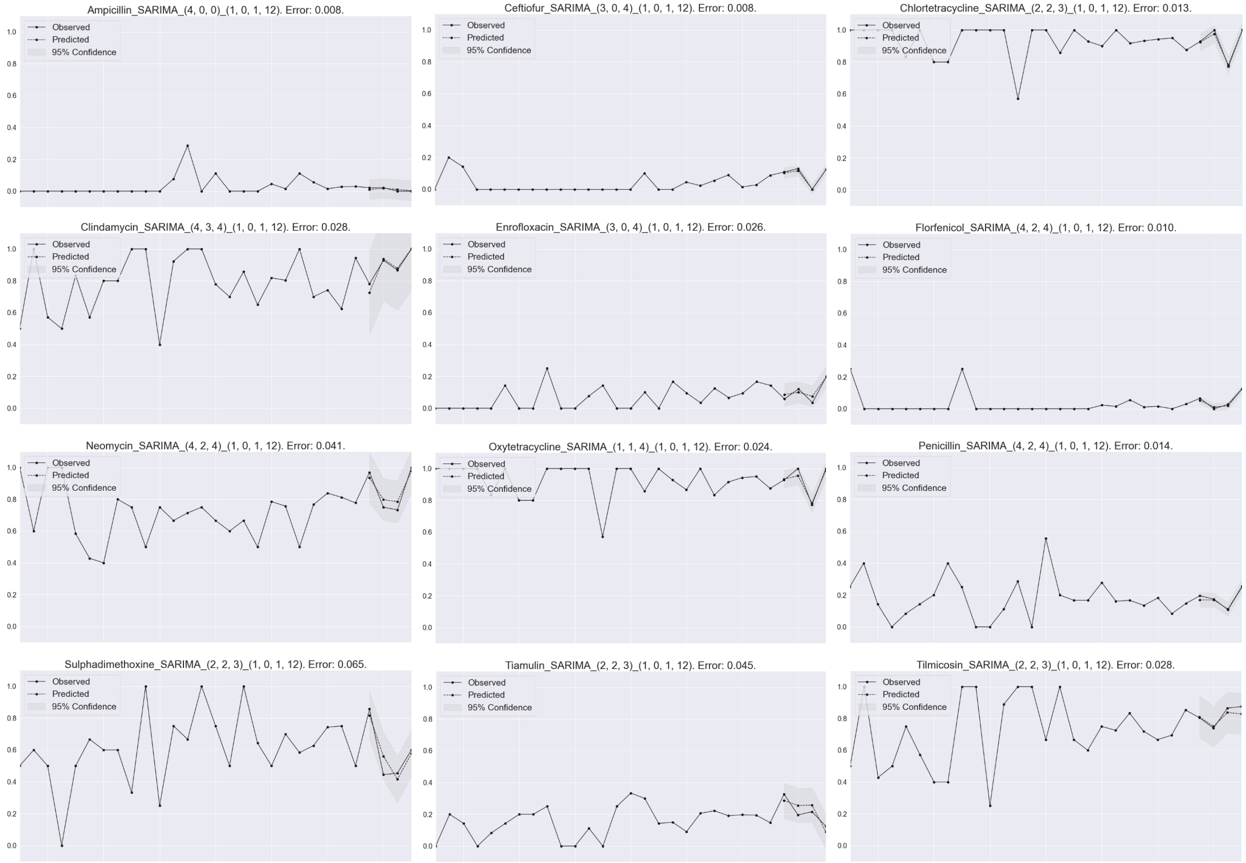
Spectinomycin_SARIMA_(4, 2, 4)_(1, 0, 1, 12). Error: 0.096.

Tiamulin_SARIMA_(4, 2, 4)_(1, 0, 1, 12). Error: 0.001.

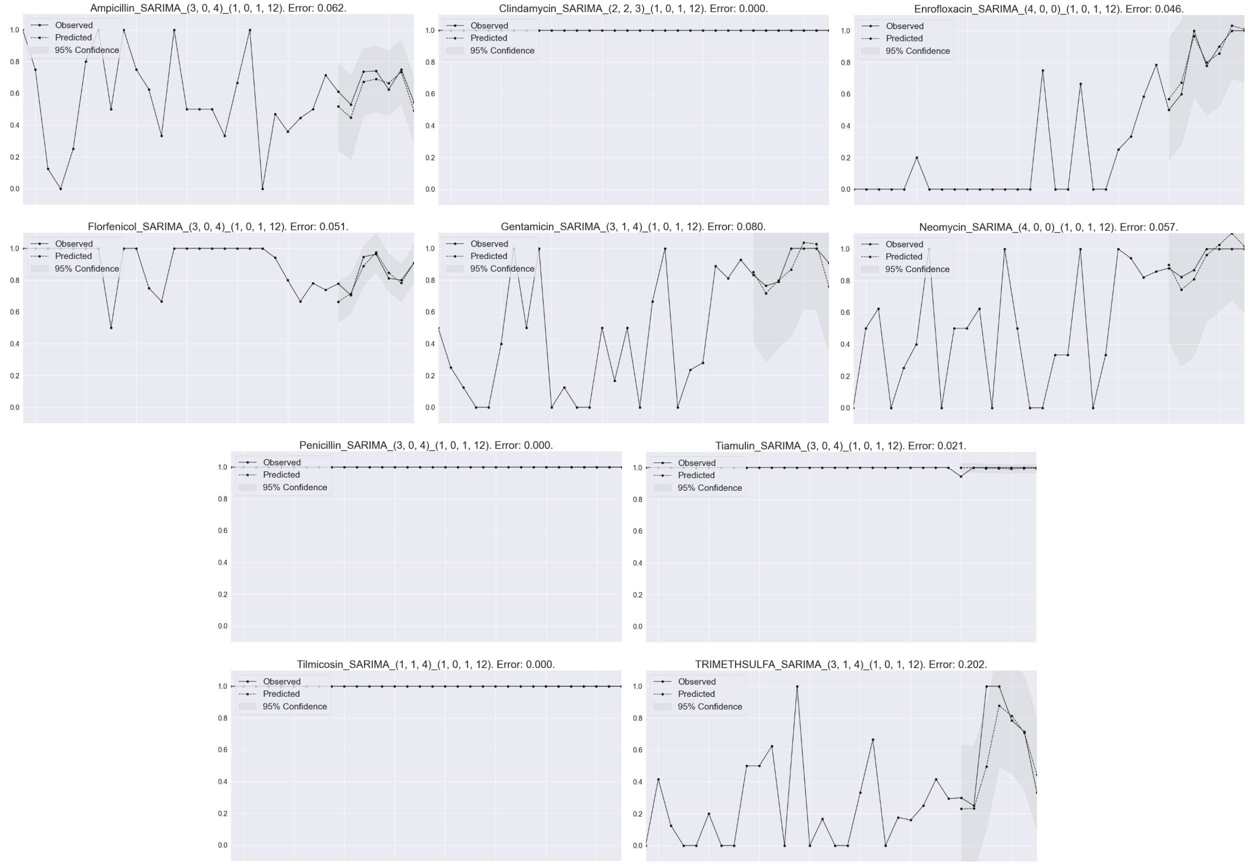
Tilmicosin_SARIMA_(2, 2, 3)_(1, 0, 1, 12). Error: 0.000.

TRIMETHSULFA_SARIMA_(4, 3, 4)_(1, 0, 1, 12). Error: 0.067.

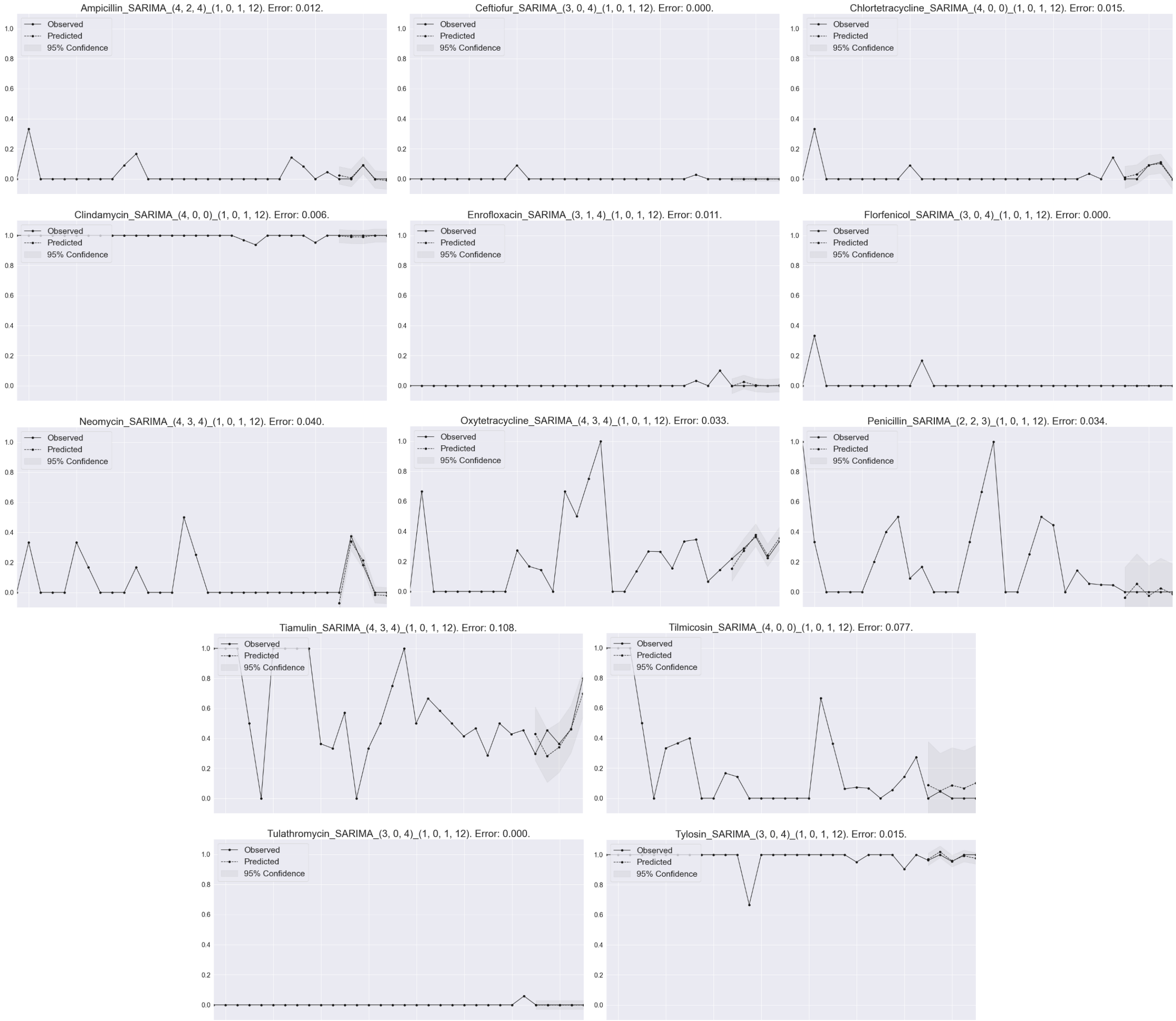
Supplement 2. Prediction results on *S. suis* time series data samples based on the SARIMA (e.g. of 9 antibiotics). Solid and dotted lines are observed and predicted values, respectively, and grey area is 95% prediction intervals.



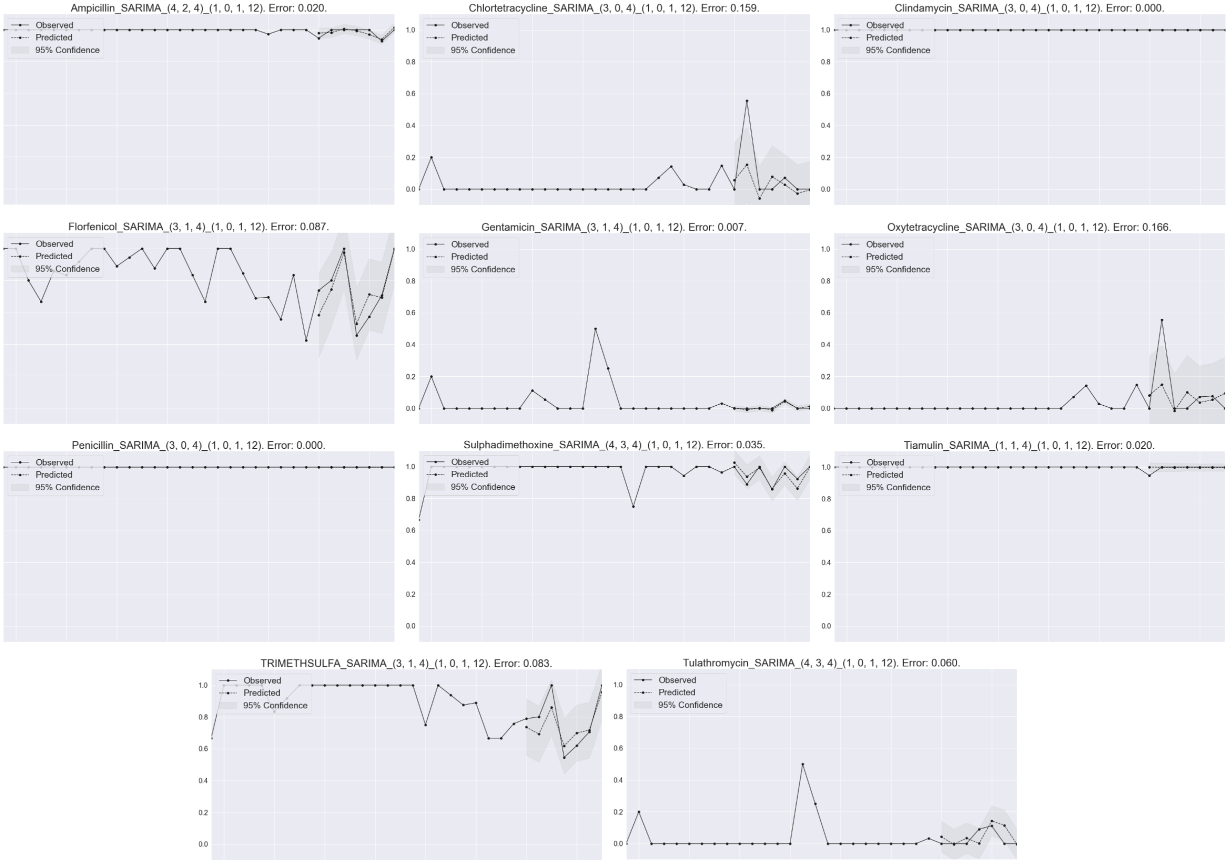
Supplement 3. Prediction results on *Salmonella sp.* time series data samples based on the SARIMA (e.g. of 9 antibiotics). Solid and dotted lines are observed and predicted values, respectively, and grey area is 95% prediction intervals.



Supplement 4. Prediction results on *Pasteurella multocida* time series data samples based on the SARIMA (e.g. of 9 antibiotics). Solid and dotted lines are observed and predicted values, respectively, and grey area is 95% prediction intervals.



Supplement 5. Prediction results on *B. bronchiseptica* time series data samples based on the SARIMA (e.g. of 9 antibiotics). Solid and dotted lines are observed and predicted values, respectively, and grey area is 95% prediction intervals.



CHAPTER 4

Predicting Minimal Inhibitory Concentration Values for 12 Antibiotics using 203 *Streptococcus suis* Whole Genome Sequences

On-going project. The methodology and results parts are shared in this dissertation.

Edited for this dissertation.

Joint work with:

Ruwini Rupasinghe, Beatriz Martínez-López
Department of Medicine and Epidemiology, Center for Animal Disease Modeling and Surveillance (CADMS), School of Veterinary Medicine, University of California, Davis, Davis, CA, United States
{rkrupasinghe, beamartinezlopez}@ucdavis.edu
Xin Liu
Department of Computer Science, University of California, Davis, Davis, CA, United States
xinliu@ucdavis.edu

4.1. Materials and Methods

4.1.1. Data Summary: Collection and Processing. Our study utilized whole genome information for 203 Streptococcus suis samples, each sample had Minimal Inhibitory Concentration (MIC) values for 21 antibiotics as summarized in Table 4.2. The sample size in our study is relatively limited compared to similar studies [**83**, **110**, **126**] and each sample has varying nucleotide lengths due to the missing information at unknown locations. The largest nucleotide length difference between the two samples was approximately a million, making data alignment for pre-processing challenging. Initial MIC values were in various forms (x, <=x, >=x, >x, <x) where x is an integer. To ensure consistency, we converted non-integer MIC values into integers following the rules outlined in Table 4.3. Furthermore, we logged MIC values with base 2, which were then utilized as expected target values for our empirical study. Additionally, each MIC value was categorized

as Resistant, Intermediate, or Susceptible. Out of 21 antibiotics in our initial data, we selected 12 antibiotics for our experiments, as breakpoints (resistant + intermediate vs susceptible) were available for these antibiotics but not for the remaining eight antibiotics. Note that samples were significantly imbalanced for some antibiotics, with most samples being either mostly resistant or mostly susceptible. For example, out of 203 samples, 192 samples were found to be resistant to Ampicillin. Throughout this paper, we used abbreviations of antibiotics as described in Table 4.1

Ampicillin	AMP
Ceftiofur	CEF
Chlortetracyline	CHL
Clindamycin	CLI
Danofloxacin	DAN
Enrofloxacin	ENR
Florfenicol	FLO
Gentamicin	GEN
Gamithromycin	GAM
Neomycin	NEO
Oxytetracyline	OXY
Penicillin	PEN
Sulfadimethoxine	SUL
Spectinomycin	SPE
Tetracycline	TET
Tiamulin	TIA
Tilmicosin	TILM
Tildipirosin	TILD
TrimethoprimSulphamethozazole	TRI
Tulathromycin	TUL
Tylosin	TYL

Table 4.1: Antibiotics name abbreviations

4.1.2. K-mer Counting. To predict phenotypic data using genotypic information, in our study, we utilized k-mer counting method [97] because 1) k-mer counting method has shown great performances in MIC prediction [83, 110, 126], 2) our current data samples do not allow for reassembling the fragments of the missing information on genome information, which requires alignment-free method that k-mer counting method falls in, 3) its sensitivity to the genomic features such as nucleotide composition, which is crucial in our situation where the data sample is very limited, and 4) its scalability by simply adjusting the k-mer size (k value) according to the desired level of resolution and analysis complexity. In our study, we explored k values of 10 and 11 to strike a

	Streptococcus suis (203 samples in total)					
	Sample size (Res./Sus.)	Res. decision (Res. + Int.)	Susceptible decision	Resistant	Intermediate	Susceptible
AMP	203 (11/192)	>=0.5	<=0.25	16	0.5, 1, 2, 4	<=0.25
CEF	203 (22/182)	>=4	<=2	8, >8	4	0.5, <=0.25, 2, <=1
CHL	32 (29/3)	>=1	<=0.5	>8	8	<=0.5, <=0.25
CLI	203 (170/33)	>=4	<=2	16, >16, 4, 8	2	<=0.25
ENR	203 (18/185)	>=1	<=0.5	0.5, 1, >2	1	0.5, <=0.12, <=0.25
FLO	203 (2/201)	>=4	<=2	-	4	0.5, 1.0, 2.0, 0.25
OXY	32 (31/1)	>=1	<=0.5	8, >8	1	<=0.5
TET	172 (165/7)	>=4	<=2	>8	4	<=0.5, 1, 2
TIA	203 (41/162)	>16	<=16	32, >32, >16	-	1, 2, 4, 8, 16, <=0.5
TILM	203 (160/43)	>16	<=16	>16, >64	-	4, 8, <=2, <=4
TILD	171 (164/7)	>=8	<=4	16, 8, >16	-	4
TUL	203 (160/43)	>=32	<=16	>64	-	<=2, <=8

Table 4.2: Initial Data summary. Data size is the number of samples used in experiments.

Conversion Rule	
>x	2*x
<x	x/2
>=x or <=x	x

Table 4.3: Conversion of non-integer MIC values.

balance between maximizing the utility of genomic information and avoiding excessive computational burden. Once the k value is selected, the occurrences of each k-mer in the genome are counted, resulting in a frequency distribution of k-mer occurrences. Subsequentially, we dropped columns where the difference between maximum and minimum counts was less than or equal to 10, based on the assumption that such corresponding columns would not significantly contribute to our final machine learning regression for MIC prediction. By doing this, we can reduce the number of features used in the regression phase as summarized in Table 4.4, resulting in saving our processing time and mitigating potential overfitting issues. Additionally, we utilized k-mer presence along with k-mer counting. Specifically, we converted all our nonzero k-mer counts into 1, while keeping all zero values unchanged. Similarly, after generating k-mer presence feature set, we dropped columns where all values were the same (either 0 or 1). As a result, we obtained four different datasets, each corresponding to two k values for k-mer counting and k-mer presence.

	K-mer counting generated	K-mer counting reduced	K-mer presence reduced
K=10	$4^10 = 128740$	128740 (12%)	57589 (5.5%)
K=11	$4^11 = 4194304$	391551 (9.33%)	379865 (9.05%)

Table 4.4: The number of columns used for each experiment. The percentage is a proportion of features used in each experiment compared to the original number of features generated by k-mer counting method.

4.1.3. Regression. In our study, we utilized machine learning regression to predict phenotypic data using genotypic information. Machine learning has been actively used in similar studies [**43**, **77**, **126**]. Among various machine learning regression methods, we employed the Random Forest (RF) method [**9**] due to several following reasons: 1) RF is an ensemble machine learning model that has shown great performances in similar studies [**34**, **127**], 2) its ability to handle high dimensional data, which is especially applicable in our case as the value of k (k-mer size) is large, and 3) its benefit of reducing overfitting issue, which is suitable for our study as we have a limited sample size and may face overfitting issue. To implement the regression, we used features extracted from k-mer counting/presence and logged MIC values as feature and target variable sets, respectively. We train and test RF by splitting samples.

4.1.4. Evaluation Metrics. To evaluate our regression results comprehensively, we used four different metrics: Mean Squared Error (MSE), Accuracy (ACC), Major Error (ME), and Very Major Error (VME). MSE is a standard machine learning regression error estimation, calculated as a mean value of squared errors between true and predicted MIC values. Lower MSE values indicate better performance. ACC is an indicator of the precision of regression results in terms of the resistant and susceptible decision as it is computed based on their resistant and susceptible decision made using predicted MIC values. A prediction is considered correct when the difference between the predicted and true MIC values is within ± 1 two-fold dilution. VME indicates the proportion of wrong prediction for resistant samples, i.e., the proportion of the resistant genomes that have been assigned susceptible MICs by the model. Food and Drug Administration (FDA) standards [**27**, **79**] for VME rates indicate that the lower and upper 95% confidence limits should be 1.5% and 7.5%, respectively. Similarly, ME indicates the proportion of wrong prediction for susceptible samples, i.e., the proportion of the susceptible genomes that have been assigned resistant MICs by the model. For this metric, FDA standards recommend a ME rate 3%.

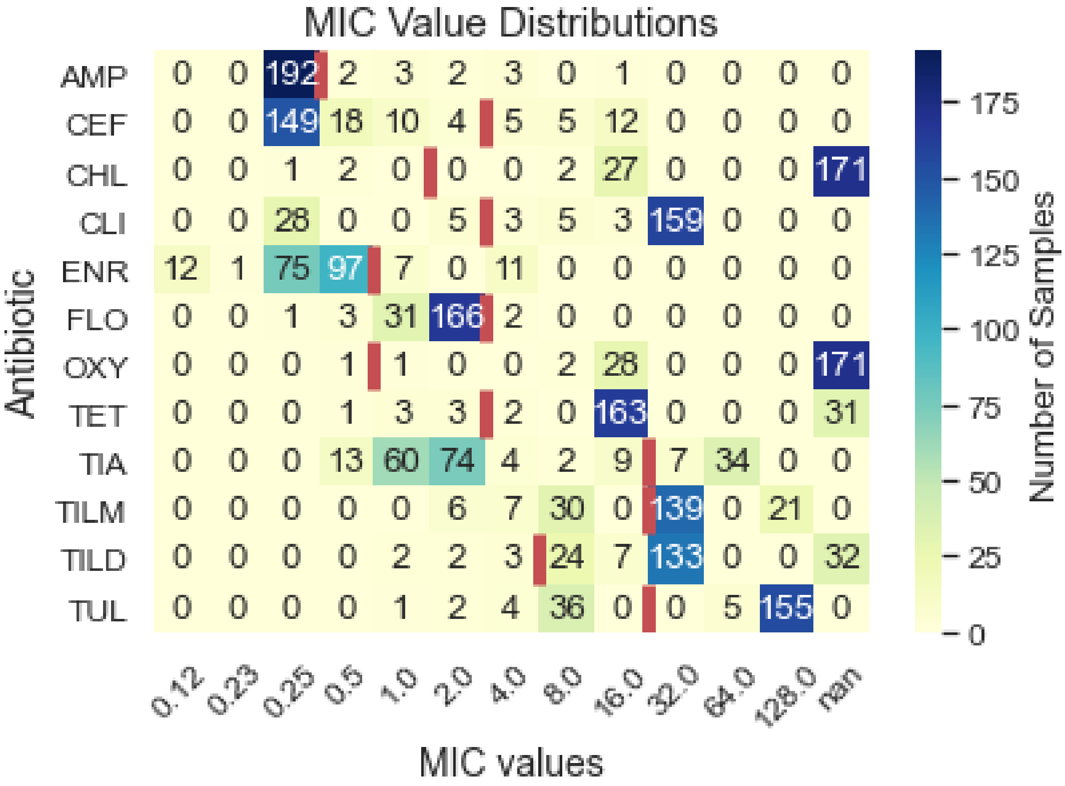


Figure 1: The number of samples for each MIC value. ‘nan’ MIC value means there is no MIC value in the data, i.e. no sample to be used for experiments. Red vertical lines are breakpoints for resistance and susceptibility. Resistant and susceptible samples are on the right and left sides of the red lines, respectively.

4.2. Numerical Results

Our numerical results include four metrics: MSE, ACC, VME and ME. To compare our results to other similar studies, we found that the proportion of antibiotics that return acceptable VME and ME is a significant measurement, as there are FDA standards for VME and ME, not for MSE and ACC considering each data from each study has its own unique features to be compared. We observed that there are a few antibiotics that do not meet FDA standards for VME and ME in any experimental setting for different k values and counting/presence as shown in Table (Perf) and Supplementary Figure 1. Specifically, CEF and TIA have imbalanced data samples biased towards susceptible samples, with 89.1% (181 out of 203) and 79.8% (162 out of 203) susceptible samples, respectively, resulting in overfitting of the model to the susceptible data and low prediction for resistant samples, leading to violation of FDA standards for VME. For ME, ME, CLI, TIA, and TUL have ME values larger than 3%, indicating a proportion of wrong predictions on susceptible data, which can be attributed to the fact that CLI and TUL have mostly resistant data samples, making it difficult for the regression model to learn information about susceptible samples. Despite these wrong predictions, our results for the proportion of antibiotics that do not meet FDA standards for VME/ME are competitive with similar studies [**83**, **110**, **126**]. Additionally, we observed that ACC values for most antibiotics are very high but CEF, CLI, TIA and TILM have relatively lower ACC outputs due to their imbalanced sample sizes as shown in Figure 2 and Figure 1. The best performance for each antibiotic is observed in both values of k and both counting/presence approaches.

4.3. Discussion

In the discussion section, we acknowledge some limitations of our study. Firstly, the inherent challenges from the dataset itself, including limited sample size (only 203 samples) compared to similar studies [**83**, **110**, **126**], diverse lengths of genome information among samples, and unknown location of missing data, which makes data alignment difficult. An imbalance in the dataset with biased samples towards either resistance or susceptibility further enhances the overfitting issue. Additionally, the breakpoints for resistant and susceptible decisions were manually determined based on our own dataset and could not be cross-validated with other resources, potentially introducing some incorrect information. Furthermore, the k-mer counting data processing approach, although showing good performance in our study, has a possible overfitting issue due to a large number of features compared to the limited number of samples, especially with larger values of k that incorporate rich information from whole genome sequences. Implementing more efficient k-mer counting methods and finding an appropriate value of k could be explored in future studies. There have been other studies that have explored optimal k-mer counting methods for similar applications [**25**, **97**]. Overall, while our study has some limitations, including the inherent challenges from the dataset itself and potential overfitting issues with the k-mer counting approach, our results are competitive with similar studies in terms of the proportion of antibiotics meeting FDA standards for VME/ME and provide insights into the performance of our regression model for antibiotic resistance prediction.

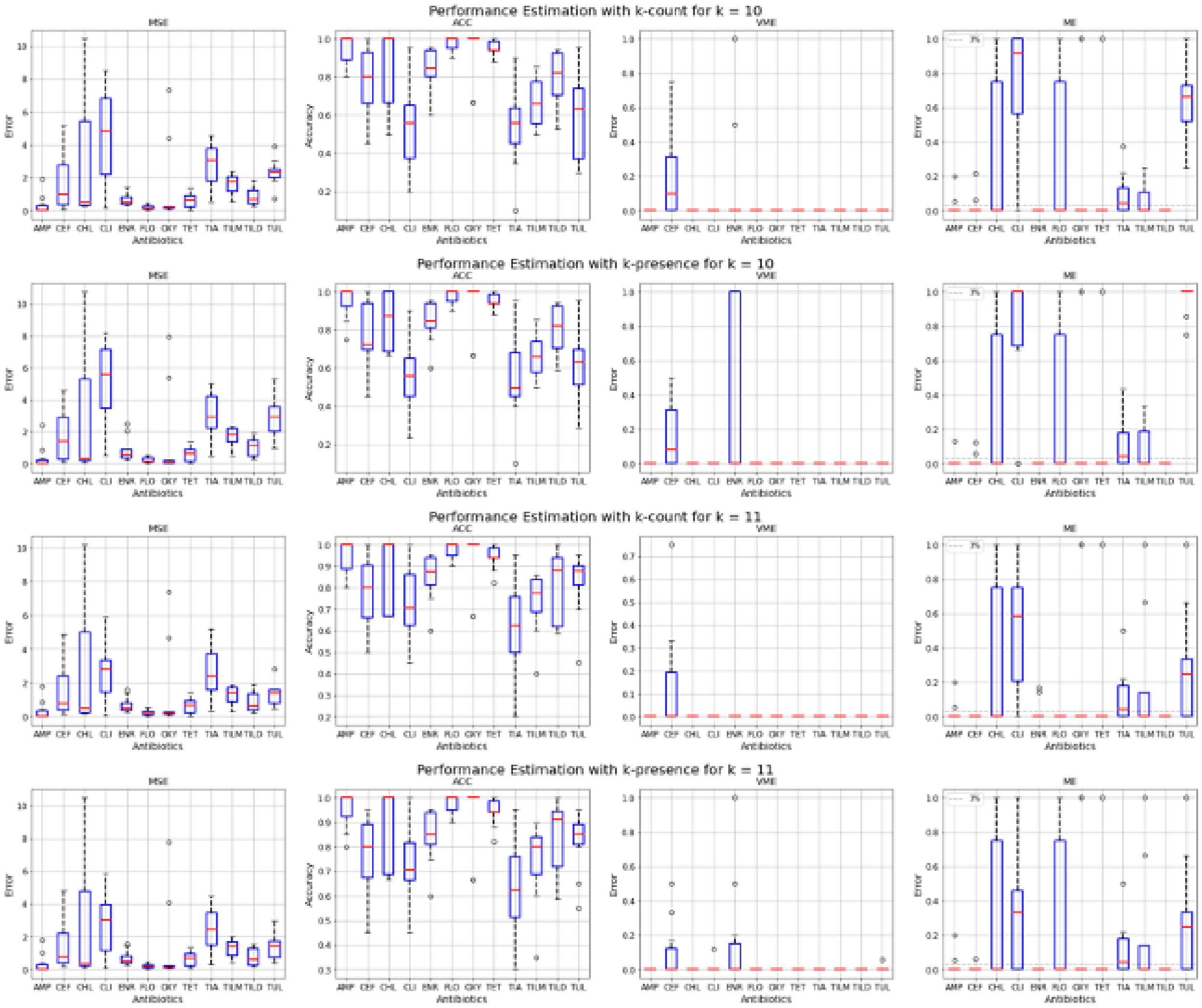


Figure 2: Overall performance estimation plot. Each represent results using either k-presence or k-count with two k values (10 and 11). Each subfigure represents different metric with detailed information including mean and standard deviation of the vales.

CHAPTER 5

OAAE: Adversarial Autoencoders for Novelty Detection in Multi-modal Normality Case via Orthogonalized Latent Space

Published in *The Thirty-Fifth Association for the Advancement of Artificial Intelligence (AAAI) Workshop on Towards Robust, Secure and Efficient Machine Learning, 2021* (Feb 2021).

Edited for this dissertation.

Joint work with:

Sungkwon An

Computational Science and Technology, Seoul National University, Seoul, Republic of Korea

skan@snu.ac.kr

Myungjoo Kang

Department of Mathematics, Seoul National University, Seoul, Republic of Korea

mkang@snu.ac.kr

Shahbaz Rezaei and Xin Liu

Department of Computer Science, University of California, Davis, Davis, CA, United States

{srezaei, xinliu}@ucdavis.edu

5.1. Abstract

Novelty detection using deep generative models such as autoencoder and generative adversarial networks mostly takes image reconstruction error as a novelty score function. However, image data, high dimensional as it is, contains a lot of different features other than class information which makes models hard to detect novelty data. The problem gets harder in multi-modal normality cases. To address this challenge, we propose a new way of measuring novelty scores in multi-modal normality cases using orthogonalized latent space. Specifically, we employ orthogonal low-rank embedding in the latent space to disentangle the features in the latent space using mutual class information. With

the orthogonalized latent space, the novelty score is defined by the change of each latent vector. The proposed algorithm was compared to state-of-the-art novelty detection algorithms using GAN such as RaPP and OCGAN, and experimental results show that ours outperforms those algorithms.

5.2. Introduction

Novelty detection, also called anomaly detection from a broader perspective, is regarded to be a task of recognizing the test data that differs in some respect from the data that are previously seen. Novelty detection has been actively researched since the demand has been increasing due to its significance and broad applications in security, AI safety, the healthcare industry.

Deep learning has recently shown tremendous performances in learning distribution and representations of various complicated data such as high-dimensional data and time series data. Deep learning for novelty detection aims to learn feature representations and output novelty scores through the neural network to detect data, which has different feature representations from the previously observed data. Many deep learning algorithms for novelty detection have been proposed recently, showing significantly better performances than traditional novelty detection methods. Deep generative models such as AutoEncoder (AE), Generative Adversarial Networks (GANs), and their variational models are recognized as one of the biggest breakthroughs in deep learning. Since they show great performances in pattern recognition in general, they are adopted for novelty detection in deep learning frameworks frequently. Deep generative models-based novelty detection algorithms such as OCGAN [**89**], RaPP [**52**], AnoGAN [**103**], [**3**], and [**102**] usually takes image reconstruction error or extension of it as a novelty score function. The key in novelty detection is to differentiate whether the input data is normal or novelty. However, as image data itself has a lot of inherent traits, e.g. rotations and thickness of the digit in images in the MNIST dataset, image reconstruction error can be magnified by those factors, which eventually increases the wrong novelty detection cases potentially as shown in Figure 1. This gets worse in the multi-modal normality case, which we aim to tackle. To the best of our knowledge, there has not been any precedent deep generative approaches to tackle novelty detection in multi-modal normality cases.

In this paper, we propose a new framework for the novelty score function using orthogonalized latent space. Detection of novelty class in latent space has several benefits. Latent space is a lower

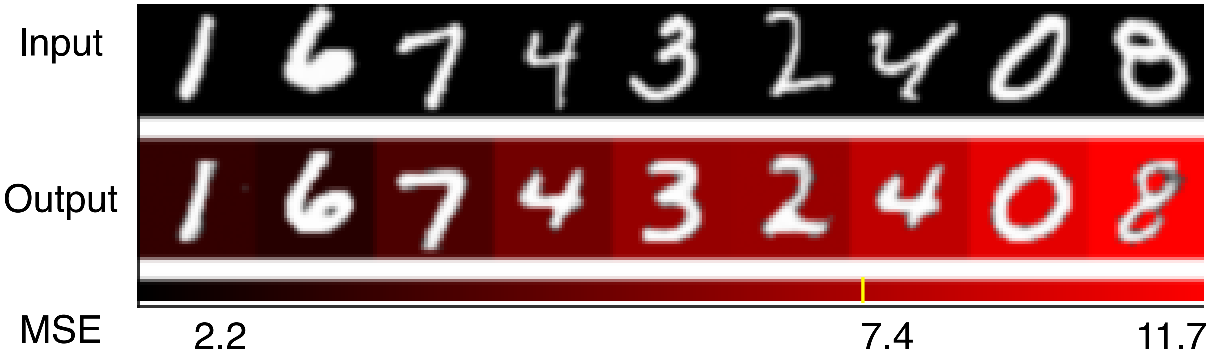


Figure 1: Limitation of novelty detection using image reconstruction error. Top: Input images. Middle: Output images of adversarial autoencoder (AAE). Bottom: mean squared error (MSE) of all images. We set the images of digits of 0-8 and 9 as normal and novelty, respectively. Since mean of novelty scores among the image of digit 9 (novelty class) is 7.4, MSE values of normal image bigger than 7.4 lead to wrong novelty detection.

dimensional space with the feature information than the original high dimensional data, which is easier to be handled. Furthermore, features in latent space can be disentangled and highlight the class information to detect novelty class well. The low dimensional trait of latent space enables us to handle the features in the data easier. In this regard, we propose a novelty function using the change of angle in latent vectors by embedding input data in latent space orthogonal to each class using mutual class information.

5.3. Related work

One-class Novelty Detection. In recent years, one-class novelty detection has received tremendous attention as a traditional representation of learning research problems. There have been many classical approaches to tackle this problem such as Principal Component Analysis (PCA). Deep learning, which has shown great performances in a variety of fields such as computer vision, cybersecurity, medical assistance, etc., finds a way to learn representation and detect the based on previously seen representation. AE-based novelty detection mostly put reconstruction errors such as mean squared error as a novelty detection function after learning the representation of the data. GAN-based novelty detection usually takes the discriminator's prediction in the image space as a tool for measuring reconstruction error. One-Class novelty detection using GAN (OCGAN) shows great performances in novelty detection in uni-modal normality data.

Approaches on Novelty Score Function. There have been other approaches to determine novelty scores other than reconstruction error or discriminator's prediction. Generative Probabilistic Novelty Detection with adversarial autoencoders (GPND) [**91**] identifies novelty data by considering it to be an inlier or an outlier. GPND has done this by utilizing a probabilistic approach and computing how likely it is that new data was generated by the normal distribution effectively. RaPP: Novelty Detection with Reconstruction along Projection Pathway (RaPP) [**52**] introduces a new way to quantify novelty scores using values in hidden space activation obtained from a deep autoencoder. RaPP compares input and its autoencoder reconstruction both in the input space and in all of the hidden spaces. However, in order to enforce their metrics, RaPP network is required to be symmetric, which makes designing network architecture and training networks a very expensive work. As the data becomes more complicated, it becomes more expensive due to fully-connected layers in the encoder and decoder caused by its structural problem. RaPP also showed a great performance in multi-modal normality cases.

5.4. Proposed Method: OAAE

In this section, We propose a new AAE novelty detection algorithm using orthogonalized latent space (OAAE) for multi-modal normality cases. The key idea is to disentangle latent space using mutual class information by employing orthogonal low-rank embedding (OLE) loss [**61**], which enables us to achieve minimizing the variance of latent vectors in intra-class as well as maximizing margins of inter-class latent vectors (in terms of angle; equivalently orthogonalize inter-class latent vectors). With such an orthogonalized latent space, we estimate a novelty score by quantifying the change of angle in each latent vector.

5.4.1. Orthogonal Latent Embedding. OLE is carried out using rank function [**61**]. Mathematical formulations of OLE begin with the following equation:

$$\arg\min_{\mathbf{T}} \sum_{c=1}^{C} rank(\mathbf{T}\mathbf{X}_c) - rank(\mathbf{T}\mathbf{X}), \text{ s.t. } ||\mathbf{T}||_2 = 1, \tag{5.1}$$

where $\mathbf{X}$ denotes the input dataset, $\mathbf{X}_c$ denotes the set of data points with class c in a subspace of $\mathbf{R}^d$, $\mathbf{T}$ is a linear transformation on the data (i.e., the feed-forward network for deep learning framework), $||\cdot||_2$ is the matrix Euclidean norm. We interpret this formulation term by term intuitively [**93**].

Minimizing the first term $\sum_{c=1}^{C} rank(\mathbf{TX}_c)$ keeps the transformed data from the same subspace a consistent representation, and maximizing the second term $rank(\mathbf{TX})$ encourages the transformed data from different subspace to represent a diverse representation. Additionally, the normalization constraint $||\mathbf{T}||_2 = 1$ avoids the trivial solution, i.e., $\mathbf{T} = 0$. Since it is known that the nuclear norm ($||\mathbf{A}||_\star$; the sum of the singular values of the matrix $\mathbf{A}$) is the convex envelop of $rank(\mathbf{A})$ over the unit ball of matrices [**26**], and due to efficiency of optimization [**10**, **95**], we reformulate the equation using the nuclear norm as follows:

$$\arg\min_{\mathbf{T}} \sum_{c=1}^{C} ||\mathbf{TX}_c||_\star - ||\mathbf{TX}||_\star, \text{ s.t. } ||\mathbf{T}||_2 = 1. \tag{5.2}$$

Following [**61**], (5.2) becomes the following loss using minibatch as below to be applied to the deep learning framework:

$$\mathbf{L}_{OLE}(\mathbf{Y}) := \sum_{c=1}^{C} \max(\Delta, ||\mathbf{Y}_c||_\star) - ||\mathbf{Y}||_\star \tag{5.3}$$

$$= \sum_{c=1}^{C} \max(\Delta, ||\Phi(\mathbf{X}_c;\theta)||_\star) - ||\Phi(\mathbf{X};\theta)||_\star. \tag{5.4}$$

To optimize (5.3) using backpropagation, the projected subgradient for the nuclear norm and the descent direction for (5.3) are obtained in by using SVD decomposition on matrix $\mathbf{A}$, i.e., $\mathbf{A} = \mathbf{U\Sigma V}^T$, and zero filling matrix $\mathbf{Z}_c$ as follows:

$$g_{||A||_\star}(A) = \mathbf{U}_1\mathbf{V}_1^T, \tag{5.5}$$

$$g_{\mathbf{L}_{OLE}}(\mathbf{Y}) = \sum_{c=1}^{C} \left[\mathbf{Z}_c^{(l)}|\mathbf{U}_{c1}\mathbf{U}_{c1}^T|\mathbf{Z}_c^{(r)}\right] - \mathbf{U}_1\mathbf{V}_1^T. \tag{5.6}$$

where $\mathbf{U}_1$ and $\mathbf{V}_1$ be the first s columns of $\mathbf{U}$ and $\mathbf{V}$, respectively, corresponding to eigenvalues larger than a small threshold value δ. Similarly, $\mathbf{U}_{c1}$ and $\mathbf{V}_{c1}$ be left and right singular vectors of $\mathbf{Y}_c$ where their corresponding singular values are greater than the threshold δ. Using $\mathbf{L}_{OLE}$ loss, we embed our high dimension dataset in orthogonalized latent space with the two main benefits: reduced variance of intra-class, maximized angle margins of clusters of inter-class as shown in Figure 2.

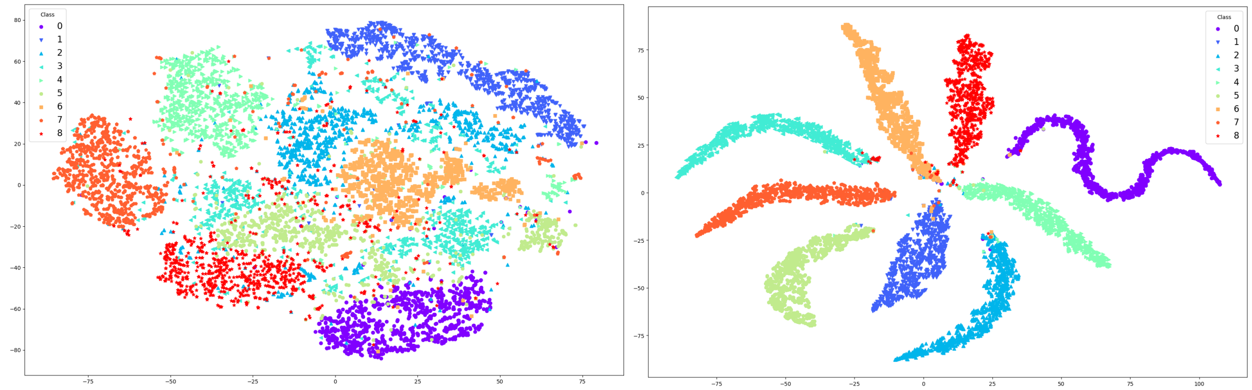


Figure 2: Embedding of trained latent space using t-SNE. Left: AAE without OLE loss. Right: AAE with OLE loss. The reduced variance of intra-class clusters of latent vectors was observed.

5.4.2. Architecture. The architecture of the proposed network is based on AAE [**71**] and a classifier was added to use mutual class information in OLE loss shown in Figure 3. Each encoder and decoder in our model has five layers with three convolutional layers and two fully-connected layers at the end. Details of training of our algorithm are described in Algorithm1. The main key in our algorithm is to adopt OLE loss to use mutual information and disentangle features in latent space and returns novelty score using the change of angles in latent vectors.

5.5. Experiment

5.5.1. Datasets. MNIST. The MNIST database, which stands for Modified National Institute of Standards and Technology database, consists of a large number of 28×28 grayscale images of handwritten digits (10 classes; 0∼9). The MNIST dataset is commonly and widely used for various computer vision and image processing research due to its simplicity. In our experiments, we choose images of one handwritten digit and every other image of the remaining nine different handwritten digits as a novelty class, and normal class data, respectively.

Fasion MNIST (f-MNIST). The fashion-MNIST is a dataset of 28×28 grayscale images 10 different classes (T-shirt; TS, Trouser; TR, Pullover; PO, Dress, Coat, Sandals, Shirt, Sneaker, Bag, Ankle boots). It shares the same image size with the original MNIST dataset but f-MNIST is regarded as a harder data to learn in general because of the complexity that semantic images have. Similar to the previous experiments on MNIST dataset, we choose images with one class (e.g.,

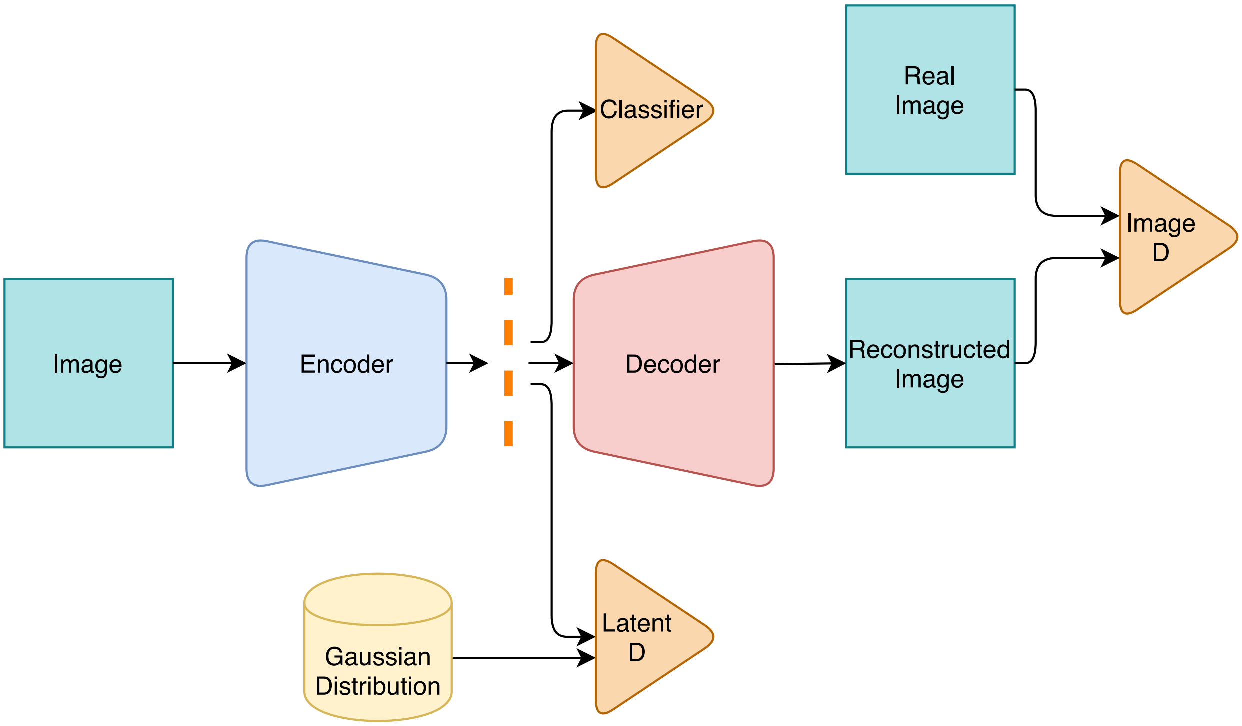


Figure 3: OAAE architecture

Table 5.1: AUROC of OAAE and the baselines

MNIST

	0	1	2	3	4	5	6	7	8	9	Mean
OCGAN	0.91	0.08	0.76	0.81	0.77	0.72	0.87	0.37	0.923	0.46	0.67
RaPP	**0.99**	0.89	**0.98**	**0.95**	0.92	**0.97**	**0.98**	0.97	0.96	0.89	0.95
OAAE	0.98	**0.97**	0.97	**0.95**	**0.95**	**0.97**	0.975	**0.97**	**0.98**	**0.97**	**0.97**

f-MNIST

	TS	TR	PO	Dress	Coat	Sandals	Shirt	Sneaker	Bag	Boots	Mean
OCGAN	0.58	0.75	0.59	0.72	0.56	0.80	0.55	0.77	0.88	0.73	0.69
RaPP	0.70	0.78	0.65	0.82	0.57	**0.85**	0.58	0.61	**0.98**	**0.82**	0.74
OAAE	**0.92**	**0.88**	**0.82**	**0.85**	**0.85**	0.72	**0.79**	**0.79**	0.97	0.79	**0.84**

CIFAR10

	AP	AM	Bird	Cat	Deer	Dog	Frog	Horse	Ship	Truck	Mean
OCGAN	0.54	0.71	0.40	0.52	0.31	0.58	0.40	0.61	0.44	0.69	0.52
RAPP	0.469	0.654	0.416	0.578	0.357	0.604	0.382	0.579	0.553	0.681	0.527
OAAE	**0.706**	**0.777**	**0.579**	**0.713**	**0.660**	**0.742**	**0.620**	**0.683**	**0.652**	**0.786**	**0.692**

T-shirt) and every other image of the remaining nine different classes as a novelty class, and normal class data, respectively.

Algorithm 1 Novelty Detection algorithm

```
**Input** : Image $x$ with class $c$, $N$ Epochs, $K$ Iteration
**Training phase**
**for** epochs 0 to $N$ **do**
    **for** iteration 0 to $K$ **do**
        $n \leftarrow \mathcal{N}(0, I)$
        $z \leftarrow \mathcal{N}(0, I)$
        Discriminator training phase
        $\mathcal{L}_{latent} \leftarrow \mathcal{D}_{latent}(z, 1) + \mathcal{D}_{latent}(Enc(x + n), 0)$
        $\mathcal{L}_{image} \leftarrow \mathcal{D}_{image}(x, 1) + \mathcal{D}_{image}(Dec(z), 0)$
        Back-propagate and update
        Encoder, Decoder and Classifier training phase
        **if** K%5 $==$ 0 **then**
            $\mathcal{L}_{recon} \leftarrow ||x - Dec(Enc(x + n))||_2^2$
            $\mathcal{L}_{Enc} \leftarrow \mathcal{D}_{latent}(Enc(x + n), 1)$
            $\mathcal{L}_{Dec} \leftarrow \mathcal{D}_{image}(Dec(z), 1)$
            $\mathcal{L}_{ole} \leftarrow OLE(Enc(x + n), c)$
            $\mathcal{L}_{cls} \leftarrow CrossEntropy(C(Enc(x + n)), c)$
            Back-propagate and update
        **end if**
    **end for**
**end for**
**Test phase**
Test image $x$
$z_0 \leftarrow Enc(x)$
$z_1 \leftarrow Enc(Dec(Enc(x)))$
$Novelty\_Score \leftarrow angle(z_0, z_1)$
```

CIFAR10. The CIFAR10 dataset consists of 60000 32×32 colored images with evenly distributed 10 classes (airplane; AP, automobile; AM, bird, cat, deer, dog, frog, horse, ship, truck). This dataset was selected due to its complexity. CIFAR10 dataset is usually treated as harder data to train than MNIST or f-MNIST in general due to its multi-channel trait.

5.5.2. Architectures of Baseline Algorithms. We compare the performance of our models to that of two state-of-the-art GANs-based novelty detection algorithms: OCGAN and RaPP. We briefly explain how those two algorithms work in the following sections.

OCGAN. OCGAN solves the classical one-class novelty detection problem and aims to determine whether the new input is from the same class or not. The key idea of OCGAN is to learn latent representations of normal class data using a denoising autoencoder network and to directly force the latent space to entirely represent the given class. OCGAN is particularly focused on learning uni-modal normality data.

RaPP. A new methodology for novelty detection is proposed in RaPP by adopting values in hidden space activation obtained from a deep AE. RAPP compares input and its AE or VAE reconstruction in the hidden spaces as well as in the input space. RaPP introduces two metrics combining those hidden activated values to measure novelty scores. In order to achieve this, RaPP requires the model to be symmetric to enforce its evaluation methodologies, which causes its structural limitation, and the training model becomes very expensive work as the data becomes more complicated due to fully-connected layers in the encoder and decoder caused by their structural problem.

5.5.3. Training Details. All of our experiments were conducted by Python 3.6.9. Adam optimizer was adopted to train our model. For stable adversarial learning, the encoder is trained with one iteration after every five iterations for the discriminator. Each experiment is carried out with 100 epochs with batch sizes as much as 64, and we set the learning rate as 0.0004. Gaussian noises with a standard deviation of 0.02 were added to the input image data during the training phase.

5.5.4. Experimental Results. We evaluate the performances of all experiments using Area Under the Receiver Operating Characteristic curve (AUROC) as shown in Table 5.1.

5.6. Discussion

Our methods showed a better performance than other previous GAN-based state-of-the-art novelty detection algorithms such as OCGAN, RaPP. Specifically, our approach provides much higher AUROC values for experiments on more complicated data such as f-MNIST, and CIFAR-10. It supports that as a tool of novelty score measurement, change of latent vector is more reasonable than image reconstruction errors since image reconstruction error can be more escalated in more complicated data. At the training level, our approach leverages class labels in a normal dataset, which is sometimes expensive work. An unsupervised learning framework without using normal class labels can be considered potentially.

5.7. Conclusion

We proposed a new novelty detection framework using deep generative models. Instead of evaluating novelty class using image reconstruction error, the change of angle in the latent vector is regarded as a tool for novelty detection quantity. We adopt OLE loss using mutual class information to achieve disentanglement of latent vectors to maximize the effect of class information. Our new approach shows a greater performance in multi-modal normality scenarios than previously existing GAN-based state-of-the-art novelty detection algorithms.

CHAPTER 6

An Empirical Study on Impact of Label Noise on Synthetic Tabular Data Generation

To be submitted in Published in *Frontiers in Artificial Intelligence* (May 2023).

Edited for this dissertation.

Joint work with:

Chao Huang and Xin liu

Department of Computer Science, University of California, Davis, Davis, CA, United States

{fchhuang, xinliu}@ucdavis.edu

6.1. Abstract

Synthetic data has been actively used for various machine learning-based tasks due to its benefits such as massive re-productivity and privacy enhancement compared to using the original data. The quality of the generated synthetic dataset crucially depends on the quality of the original data, which, in practice, is usually, corrupted by label noise. While there have been studies on feature noise, how label noise affects synthetic data generation is under-explored. In this paper, we evaluate the impact of the label noise label on synthetic data generation with a focus on tabular data. One challenge is how to evaluate the quality of synthetic data under label noise. To this end, we design comprehensive experiments to measure the impact of label noise on synthetic data generation in different aspects: synthetic data quality, data utility, and convergence for training synthesizers and machine learning models for downstream tasks. The empirical results cover wide aspects of synthetic data generation under label noise and they show quality and utility degrades with higher noise levels while there is no significant effect on the synthesizer convergence observed.

6.2. Introduction

Synthetic data has become an increasingly popular choice for machine learning training due to its advantages such as potential massive data re-productivity and privacy preservation compared to the original data. It has been used for many practical applications such as healthcare/medicine [**14**, **20**] and manufacturing [**85**]. In practice, synthesizers can generate various data types such as image, time series data, and tabular data. In this work, we focus on tabular data generation because tabular data is quite prevalent in many practical scenarios, such as electronic health records [**38**], patient data [**6**], and hiring/loan decisions [**41**], and also tabular datasets have more diverse data types for features such as categorical and numerical features while image data has only numerical values. Generating useful quality synthetic datasets is largely dependent on the quality of the training data for synthesizers. In this regard, it is significant to investigate the factors affecting the quality of training data for synthesizers and how those factors influence synthetic data generation eventually.

Label noise has been extensively studied in literature due to its critical effect on machine learning tasks, and it has been shown that label noise degrades machine learning performance in downstream tasks. While it is important to consider label noise for synthetic data generation, it has not been well investigated enough, especially on the tabular datasets. There have been a few studies that discuss the vulnerability of synthesizers on label noise [**48**, **112**], which, however, is limited to image datasets. To our best knowledge, there has been no such work to comprehensively investigate the impact of label noise on synthetic tabular data generation and quantify their effects. Our work aims to fill this gap.

Previous studies showed from a rather coarse perspective that label noise lessens model performance. [**45**]. This paper attempts to proceed from a much more systematic and finer-grained perspective on how label noise affects synthetic tabular data generation. We are particularly interested in answering the following research questions:

- Question 1: How does label noise affect the convergence rate for training synthesizers?
- Question 2: How does label noise affect the quality, e.g., fidelity/diversity/generalization power, of the synthetic tabular data?
- Question 3: How does generated synthetic data affect downstream tasks? More specifically,
  · How useful can generated synthetic tabular data be as training data for downstream tasks?

· How does generated synthetic data affect downstream machine learning models such as training and validation performance?

To answer Question 1, we conduct numerical experiments on training synthesizers using training data with generated label noise. Interestingly, we found that the effects of label noise on synthesizer convergence are not significant such as overfitting or convergence failure, regardless of the noise level. To answer Question 2, we conduct numerical experiments to measure the quality of generated synthetic data using several metrics (which will be discussed in detail later in Section 3.3 B). Our choice of metrics enables a comprehensive understanding of the quality of generated data. Our results show that the quality of synthetic data decreases with the higher noise levels. We also observed that two different label noise generation approaches affect the synthetic data quality a bit differently. Even though there is a clear quality degradation of synthetic data with the higher noise level, the qualities of generated data are still high compared to the real data. Finally, to answer Question 3, we conduct numerical experiments on training machine learning models on classification tasks using generated synthetic data and testing them on real data. We found that synthetic data utility and the performance of machine learning models decrease with higher levels of label noise. We observe relatively high variances in synthetic data utility for the two label noise generation approaches used. Additionally, we observe that synthetic data itself caused an overfitting problem for downstream machine learning tasks and larger noise levels outputted a high gap between training and validation accuracies.

The key contributions of this paper are summarized below:

- To the best of our knowledge, this is the first comprehensive empirical study that analyzes the impact of label noise on synthetic tabular data generation. Our empirical study reveals important implications for the use of synthetic data in machine learning applications, particularly when dealing with noisy or unreliable labels.
- Numerical experiments are systemically conducted with different synthesizer algorithms and different label noise generation settings with various levels of noise. As the evaluation of the quality of the generated data is challenging, we utilized multiple metrics to touch on it in various aspects.

- Our study provides several interesting observations and insights: 1) label noise generally degrades in quality and utility of generated synthetic tabular data, 2) interestingly, label noise has little impact on the convergence of training synthesizers, 3) sometimes we observed that there is performance increase at random places with higher levels of noise, 4) synthetic data generated from larger noise levels cause larger gaps in training and validation accuracies for the downstream task, and 5) there is no seemingly significant differences in performances between different label noise generations.

The remaining of the paper is organized as follows. In Section 2, we provide the preliminaries. In Section 3, we present our experiments. In Section 4, numerical results are provided. In Sections 5 and 6, we discuss and conclude our study.

6.3. Preliminaries

6.3.1. Synthetic Tabular Data Generation. Generative Adversarial Networks (GAN) [**31**], a class of deep learning models, has been extensively used for synthetic data generation. GAN consists of two neural networks: a generator and a discriminator. The generator learns to produce synthetic data that is similar to the real data, while the discriminator learns to distinguish between real and synthetic data. The two networks are trained simultaneously in a min-max game process. The generator is expected to produce diverse data that the discriminator cannot easily identify from real data. Additionally, the training process is known to be able to be unstable, resulting in low quality synthetic data and failure of model convergence [**54**].

Among many variations of GAN models, we discuss synthetic tabular data generation algorithms relevant to our study. CTGAN (Modeling Tabular data using Conditional GAN), TVAE (VAE-based Deep Learning data synthesizer) [**125**], CopulaGAN (Apply Copula flow [**47**] to CTGAN). GAN-based methods have shown great performances in generating synthetic data in general, and especially these three methods are specialized in generating tabular data with both categorical and numerical columns. To be specific, CTGAN, one of the most frequently used synthetic tabular data generation methods, uses a conditional approach to GAN [**31**] to generate the tabular data more realistically. Similar to CTGAN, TVAE utilizes VAE (Variational Auto Encoder) structure and generates synthetic tabular data. CopulaGAN apply Gaussian copulas [**47**] to CTGAN to generate high-fidelity synthetic data. These three models are frequently used in synthetic tabular data generation algorithms as baselines in recent literature [**8**] as they share a similar structure but have shown different performances.

While GAN-based synthesizer has shown a great performance in synthetic data generation, we face a few challenges [**58**] such as overfitting and handling diverse data types, especially for tabular datasets. For this reason, it is significant to study the convergence of the synthesizer and the quality of generated data in synthetic data generation. Recently, there have been GAN-based data generation studies that tackle feature noise in data [**94**, **100**]. There is no study on the systemical impact of label noise on synthetic tabular data generation. Our study aims to fill in this gap through empirical experiments touching on different aspects of the influence.

6.3.2. Noisy Data Generation. In our experiments, we have a label with binary classes to work with and consider label corruption from one label to the other label. In this section, we discuss two different ways to generate noisy labels used in our study: random flipping (instance-independent) and instance-dependent label noise. We describe the noisy label generation process in Figure 1.

A. Random flipping (instance-independent) noise label generation.

Random flipping noise, an instance-independent noise, is a benchmark where a label is flipped to another class at random [**32**]. It does not take into account the feature sets. More specifically, Each label is flipped into the other class with a probability p_i. Workflow with this instance-independent random flipping label generation method is described in Figure 1.

B. Instance-dependent noise label generation.

Instance-dependent noise is a more realistic noise type that incorporates feature sets. To do this, we used a pre-trained machine learning model to classify the samples and followed the decision made. The model simulates how human generates noisy labels. From there, we collected the set of samples where the predicted label is not identical to their original label. From the previously obtained set, we randomly select samples with a probability that we actually flip the label with different noise levels (10% - 50%). Workflow with instance-dependent label generation method is described in the top Figure in Figure 1. It has been used actively in literature in various domains [**117**, **124**, **129**]. This approach is more targeted and is able to generate more consistent label noise in data for multiple generation attempts than random flipping.

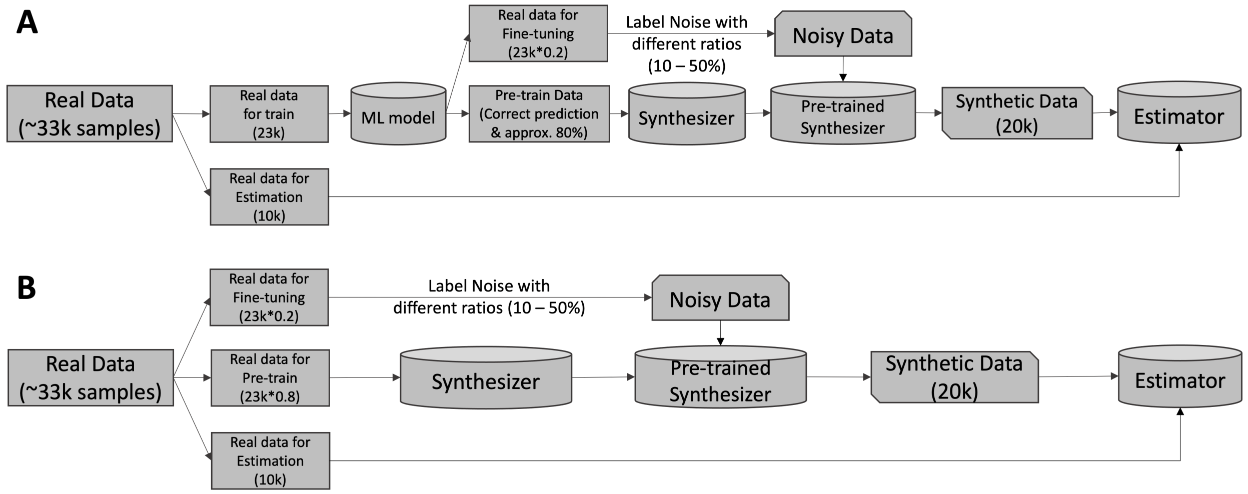

A
Real Data (~33k samples)
Real data for train (23k)
ML model
Real data for Fine-tuning (23k*0.2)
Pre-train Data (Correct prediction & approx. 80%)
Label Noise with different ratios (10 – 50%)
Noisy Data
Synthesizer
Pre-trained Synthesizer
Synthetic Data (20k)
Estimator
Real data for Estimation (10k)
B
Real Data (~33k samples)
Real data for Fine-tuning (23k*0.2)
Real data for Pre-train (23k*0.8)
Real data for Estimation (10k)
Label Noise with different ratios (10 – 50%)
Noisy Data
Synthesizer
Pre-trained Synthesizer
Synthetic Data (20k)
Estimator

Figure 1: Workflows with two different label noise generation approaches. (A) Random flipping noise. (B) Instance-dependent noise

6.4. Experiments

6.4.1. Dataset. To assess the impact of label noise on the synthetic tabular data generation, we utilize Adult data from the UCI dataset repository [**24**]. Adult data consists of approximately 32000 samples, each with 14 attributes (columns or features), including nine categorical and four numerical columns. Our downstream machine learning task is to classify the adult tabular data and determine whether the income level is more than $ 50,000/year or not, based on the remaining 13 attributes. A few samples are described in Table

Sample #	**age**	**education-num**	···	**education**	**race**	**sex**	···	**income**
1	39	13	···	Bachelors	White	Male	···	<=50K
2	50	13	···	Bachelors	White	Male	···	<=50K
3	38	9	···	HS-grad	White	Male	···	<=50K
4	53	7	···	11th	Black	Male	···	<=50K
5	28	13	···	Bachelors	Black	Female	···	<=50K

Table 6.1: Five data samples in UCI Adult dataset with attributes (bold columns names). Two (out of six) continuous columns and four (out of nine) categorical columns including binary class annual income column are presented.

6.4.2. Experimental Workflow. Our experiments aim to examine the impact of label noise on synthetic tabular data generation and involve three main components: (1) generating data with varying levels of label noise, (2) training synthesizers and using them to generate synthetic tabular data and (3) evaluating the quality of generated data. To carry out these experiments, we divided our samples into two sets: a training set with 22k samples, which we used to train the synthesizer, and an evaluation set with 10k samples, which we used to assess the utility of the synthetic tabular data.

A. Label noise generation.

To evaluate the impact of label noise on synthetic tabular data generation, we use two methods: random flipping and instance-dependent noise generation. For random flipping, we flip the label of each sample in the training set with a specified probability (i.e., noise level) ranging from 0% to 50% For instance-dependent noise generation, we first train a logistic regression classifier (binary classification for annual income; over 50K or not) on the 23k samples. We then identify all samples with different predicted outputs compared to their original labels, and randomly select 4600 samples, a subset of roughly 20% of the training samples, as candidates for label flipping. From this group, we randomly select samples with varying noise levels ranging from 10% to 50% similar to random flipping.

B. Training and generating the synthetic data.

Training synthesizers from scratch using noisy labels can be very unstable so it is essential to stabilize the synthesizer. Also, we want to minimize the impact of synthesizers (deep generative models), e.g., training instability, and focus on how the label noise affects synthetic data generation. To this end, we first pre-train the synthesizer with clean data (approximately 18400 samples; 80% of training data (22k)). After pre-training the synthesizer, we fine-tune the model with noisy data with different noise levels (0% - 50%) [**37**]. With the trained generator component of the trained synthesizer, we generate the synthetic tabular adult dataset with 20k samples.

C. Synthetic data evaluation.

To comprehensively measure the impact of label noise on synthetic tabular data generation, we evaluate it from various aspects, which will be detailed in Section 3.3.

6.4.3. Evaluation Metrics. To comprehensively understand the impact of label noise on synthetic tabular data generation, our study uses various metrics. We first provide an overview of the metrics and then discuss them in detail.

- Convergence analysis: synthesizer training and downstream classification task.

  · Generator and discriminator loss.

  · Learning curve for machine learning methods.

- Data quality: how good synthetic data is compared to real data.

  · Statistical Similarity with two different methods: Chi-squared (CS) test and Total Variation (TV).

  · Sample-level similarity [1]: fidelity, diversity, and generalization.

  · Label accuracy: measure how correctly synthetic labels are generated with label noise.

- Data utility: how well synthetic data can be used as training data for downstream tasks.

  · Machine learning efficacy.

A. Convergence analysis.

For convergence analysis, we observe the behaviors of generator and discriminator loss functions for synthesizers and learning curves for downstream machine learning models. It has been observed that label noise causes harmful effects on training neural networks such as overfitting [23, 50] and convergence failure [70]. This motivates us to study how label noise affects the convergence of training synthesizers. First, to evaluate the convergence rate for the training synthesizer, we observe the loss function behavior for both the generator and the discriminator in GAN-based synthesizers and analyze the stability and convergence rate in the training phase. Similar to this, we observe learning curves to estimate the training stability, performance saturation, and under/overfitting for downstream machine learning models.

B. Data quality.

To assess this part more comprehensively, we adopt three different aspects as follows.

1) Statistical similarity. This metric measures how similar the generated synthetic data is to the real clean data statistically. To this end, we use three different column-wise distribution comparison methods: Chi-squared (CS) test [88], Total Variation (TV) [29] and the Kullback–Leibler(KL) Divergence [18]. For the CS test, averaged p-value between 0 and 1 across the columns is outputted using the equation below:

$$CS(R_i, S_i) = \chi^2(R_i, S_i) = \sum_k \frac{(R_i^k - S_i^k)^2}{S_i^k} \text{ for } 1 \leq i \leq 15. \tag{6.1}$$

where R and S are real and synthetic data, respectively. R_i^k and S_i^k are the kth element of ith column for real and synthetic tabular data, respectively. Using the distribution of χ, we obtain a p-value and an averaged p-value between 0 and 1 across the columns. A higher p-value represents high similarity to the real data. TV method returns the averaged value of (1-(TV distance)) across the columns using the following equation.

$$TV(R_i, S_i) = \frac{1}{2} \sum_{c \in C_i} |Freq_c(R_i), Freq_c(S_i)| \text{ for } 1 \leq i \leq 15. \tag{6.2}$$

where c that represent all possible categories in a column set C_i and $Freq$ is a frequency count function for c in either R_i or S_i. Note that a higher score indicates higher similarity. An averaged KL divergence value across all columns is returned after normalization following the equation below.

$$KL(R_i, S_i) = \sum_i Dist_{R_i} ln \frac{Dist_{R_i}}{Dist_{S_i}} \text{ for } 1 \leq i \leq 15. \tag{6.3}$$

where $Dist$ represents a distribution of each column. We conducted 10 runs and reported the mean/standard deviation of all runs, and for each run, we generated 20k synthetic samples and compute the similarities.

2) Sample level analysis. A 3-dimensional evaluation metric named (α-Precision, β-Recall, Authenticity), is proposed [**1**] where α-Precision, β-Recall are a generalization of precision and recall, respectively. The metric provides information on fidelity, diversity, and generalization, respectively. Fidelity here indicates how realistic the synthetic samples are. Diversity value represents how diverse the generated outputs are, i.e., a variety of generated samples. Generalization indicates whether generated samples are direct copies of the real data to measure overfit of the model by observing the probability that synthetic samples are duplicated from the real training data. This metric has been used in many recent studies and measures the quality of the synthetic data. [**63**]

3) Label accuracy. This metric measures how accurately synthesizers generated synthetic labels. To this end, we train machine learning models using original data and test them on generated synthetic data. This metric is usually adopted in the synthetic image generations [**113**] because synthetic image generation has a separate feature set (image) and label. In our case, this is also a valid metric because machine learning models trained on synthetic data can be very unstable which can result in high variance output as inconclusive results.

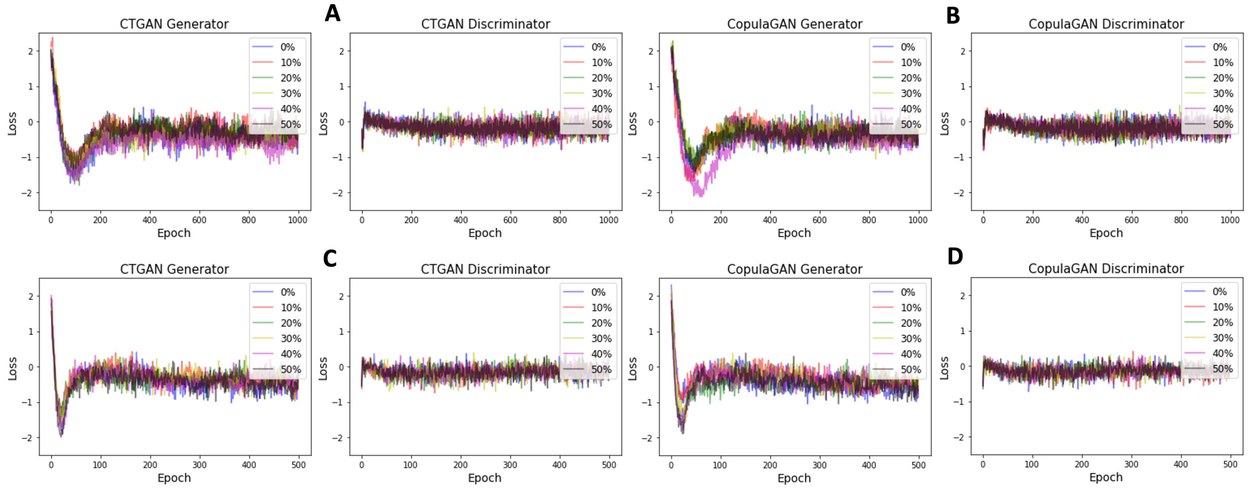


Figure 2: Generator and discriminator loss behavior analysis for GAN-based synthesizers: CTGAN and CopulaGAN with two noise label generation approaches. (A) CTGAN with random flipping noise. (B) CopulaGAN with random flipping noise. (C) CTGAN with instance-dependent noise. (D) CopulaGAN with instance-dependent noise.

C. Data utility.

We adopt machine learning efficacy to measure data utility. This metric measures how effectively generated synthetic data can be used as training data for downstream machine learning tasks because synthetic data is often used to train machine learning models before they are tested on the real data. In our experiments, we train various machine learning models (Logistic Regression; LR, Decision Tree; DT, random Forest; RF and Multi-Layer Perceptron; MLP) on synthetic samples and test on real data for classifying adult data into binary classes (annual income >50k vs <=50k). To conduct this, we generated 20k synthetic samples 10 times independently from trained synthesizers and used them for training downstream machine learning models to evaluate accuracy on real 10k samples.

6.5. Numerical Results

We present experiments to study the impact of noisy labels on synthetic tabular data generation from different perspectives: convergence rates for training synthesizer (Section 4.1), synthetic data quality (Section 4.1; similarity and label accuracy), downstream classification machine learning tasks (Section 4.3; utility and learning curves of machine learning models ()).

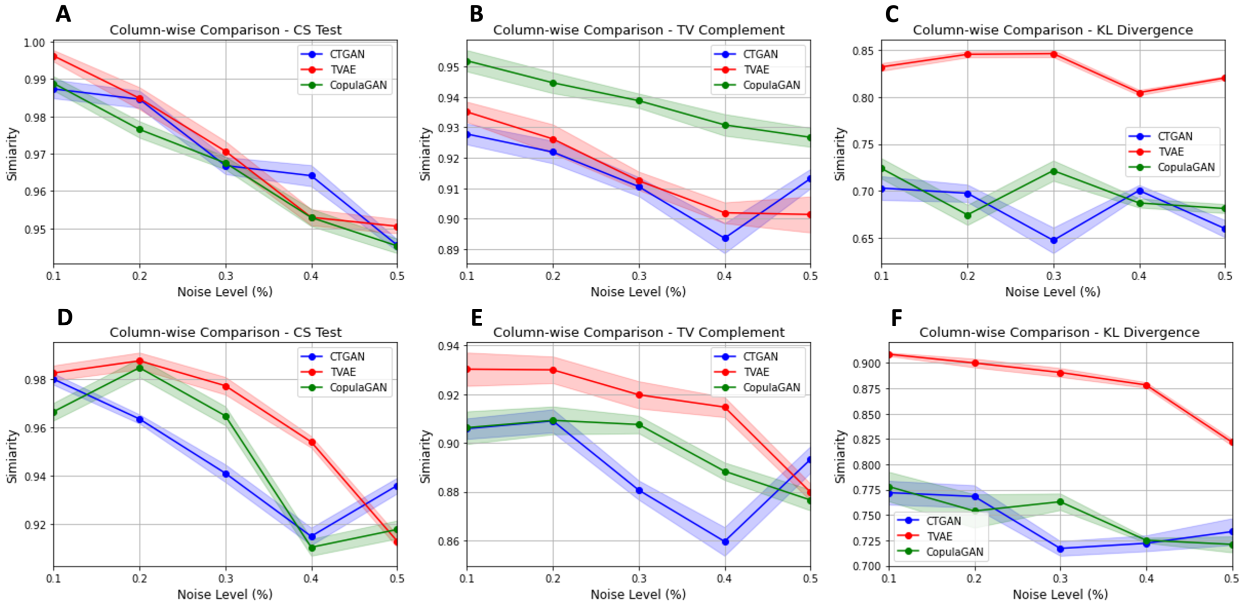


Figure 3: Three statistical similarity comparison metrics plots (CS Test, TV Complement and KL Divergence) with two different noise label approaches. (A) CTGAN with random flipping. (B) TVAE with random flipping. (C) CopulaGAN with random flipping. (D) CTGAN with instance-dependent. (E) TVAE with instance-dependent. (F) CopulaGAN with instance-dependent.

6.5.1. Impacts on the convergence for training synthesizer. Empirical results on behaviors of training synthesizers are presented in Figure 2. Our findings observed that there is no significant delay in training synthesizer with higher noise levels even though overfitting and convergence failure are observed in some similar studies [**23**, **50**, **70**]. With higher label noises, the generator and discriminator loss converge similarly to the lower noise levels. For the CopulaGAN model with random flipping noise type, there is a visible delay when the noise level is 40%, but it is still not a significant delay compared to other noise levels. Note that this can be explained that our loss plots are for fine-tuning with noisy data after pre-training and it is possible that the model is stabilized enough in the pre-training phase with the original data as in similar studies such as pre-trained language model [**123**], it is observed that fine-tuning the model with a little noise after pre-training can help the stabilizing the model better.

6.5.2. Synthetic data quality. Empirical results on data quality of synthetic data are presented in Figure 3. We observed that statistical similarity between the generated synthetic data and original data decreases relatively coherently in terms of p-values from the CS test, TV complements, and

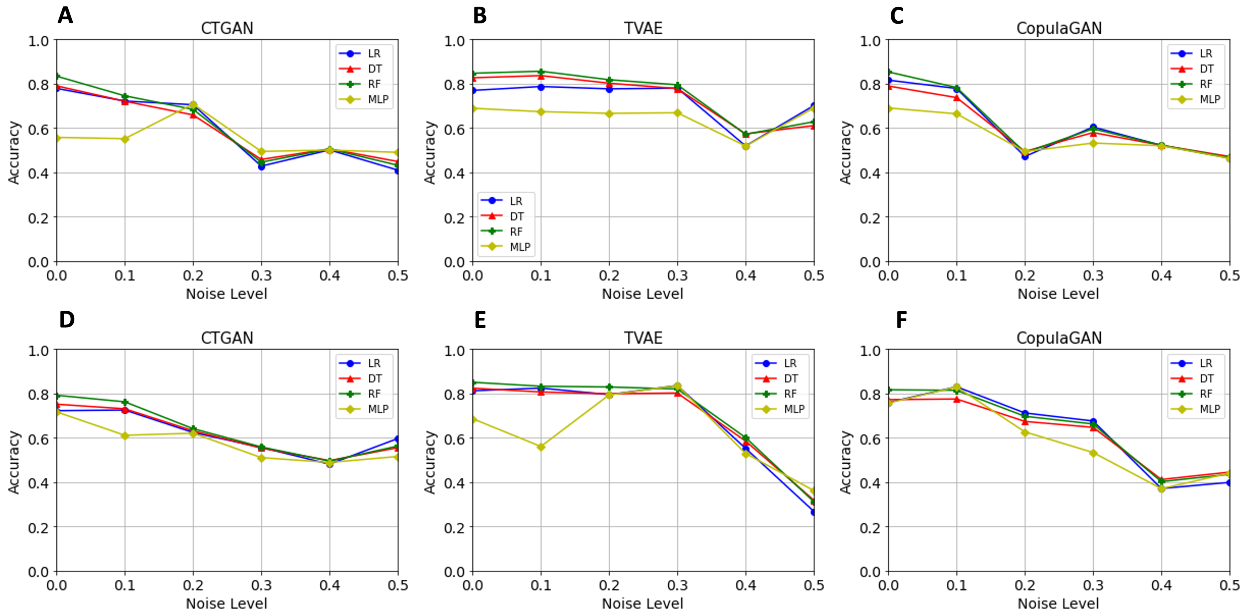


Figure 4: Label accuracy (mean value from 10 runs) for three synthesizers with two different noisy label generation methods. (A) CS test with random flipping. (B) TV Complement with random flipping. (C) KL Divergence with random flipping. (D) CS test with instance-dependent. (E) TV Complement with instance-dependent. (F) KL Divergence with instance-dependent.

KL divergence values. There are a few cases where similarity increases for larger noise levels, e.g., level 40% to 50% for CTGAN for instance-dependent noise type. This has been observed in other similar studies [**49**, **87**, **112**]. Note that this happens rarely and overall behavior has decreasing trend in general. We also observe that random flipping label noise affects the quality degradation more linearly in general compared to the instance-dependent label noise approach. Specifically, the CS test and TV complement, with instance-dependent label noise, decrease more dynamically compared to the random flipping noise.

Additionally, synthetic label accuracy decreases as higher noise levels are used as shown in Figure 4. More comprehensive results including the standard deviation values are described in Supplementary Table 6.3. Compared to the standard deviation results from machine learning efficacy estimation (Supplementary Table 6.2), it shows relatively small variances. This can be explained that the machine models are trained with real clean data, which makes the results more stable than training with synthetic data.

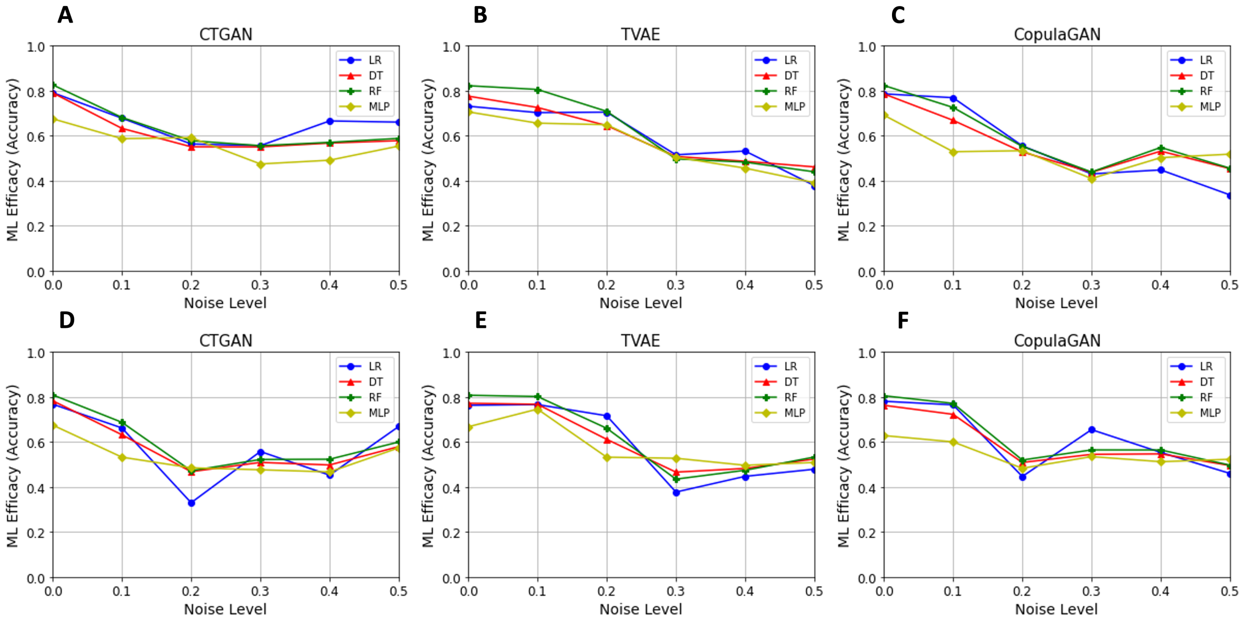


Figure 5: Machine learning efficacy (mean value from 10 runs) for three synthesizer with two noise label generation approaches. (A) CTGAN with random flipping. (B) TVAE with random flipping. (C) CopulaGAN with random flipping. (D) CTGAN with instance-dependent. (E) TVAE with instance-dependent. (F) CopulaGAN with instance-dependent.

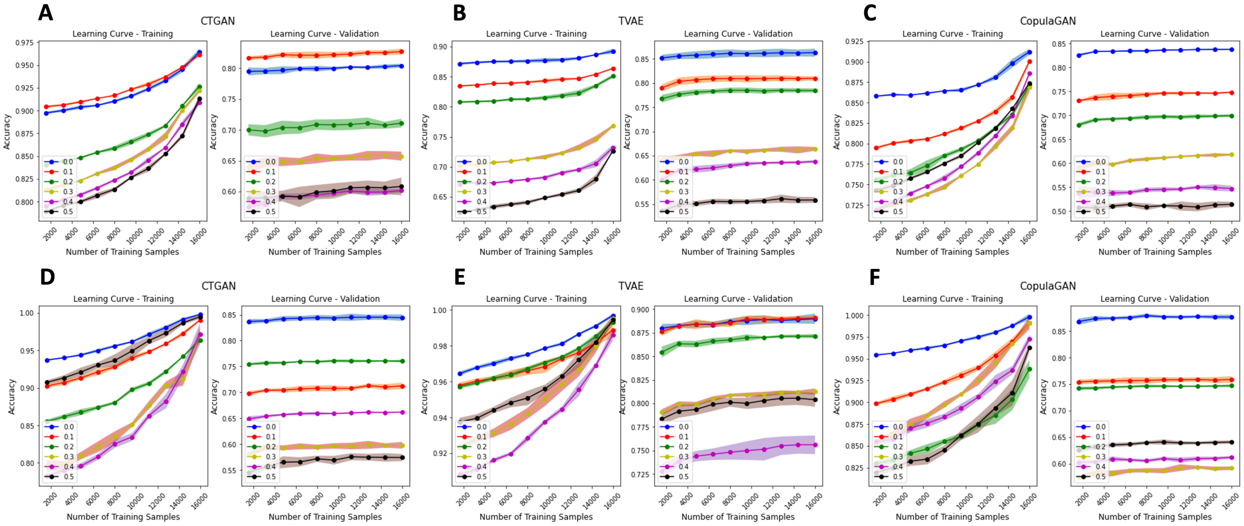


Figure 6: RF learning curves (training curve and test curve) for three synthesizers with two noise label generation approaches. (A) CTGAN with random flipping. (B) TVAE with random flipping. (C) CopulaGAN with random flipping. (D) CTGAN with instance-dependent. (E) TVAE with instance-dependent. (F) CopulaGAN with instance-dependent.

Random Flipping Label Noise

2*Synthesizers	2*ML models	Noise Level					
		0.0	0.1	0.2	0.3	0.4	0.5
4*CTGAN	LR	0.79/0.01	0.68/0.22	0.56/0.28	0.56/0.29	0.67/0.24	0.66/0.24
	DT	0.79/0.0	0.63/0.08	0.55/0.13	0.55/0.11	0.57/0.08	0.58/0.12
	RF	0.83/0.01	0.68/0.11	0.58/0.16	0.56/0.16	0.57/0.08	0.59/0.15
	MLP	0.68/0.09	0.59/0.11	0.59/0.13	0.47/0.08	0.49/0.06	0.55/0.05
4*TVAE	LR	0.73/0.01	0.7/0.03	0.7/0.07	0.52/0.19	0.53/0.24	0.38/0.04
	DT	0.78/0.03	0.73/0.01	0.64/0.08	0.51/0.04	0.49/0.07	0.46/0.06
	RF	0.82/0.01	0.81/0.0	0.71/0.09	0.5/0.1	0.48/0.1	0.44/0.1
	MLP	0.71/0.04	0.66/0.06	0.65/0.03	0.5/0.09	0.46/0.13	0.39/0.07
4*CopulaGAN	LR	0.79/0.01	0.77/0.0	0.56/0.29	0.43/0.27	0.45/0.29	0.34/0.24
	DT	0.79/0.01	0.67/0.07	0.53/0.13	0.44/0.13	0.53/0.13	0.45/0.09
	RF	0.82/0.0	0.73/0.07	0.55/0.17	0.44/0.14	0.55/0.14	0.46/0.11
	MLP	0.69/0.03	0.53/0.16	0.53/0.12	0.41/0.1	0.5/0.06	0.52/0.1

Instance-dependent Label Noise

2*Synthesizers	2*ML models	Noise Level					
		0.0	0.1	0.2	0.3	0.4	0.5
4*CTGAN	LR	0.77/0.01	0.66/0.24	0.33/0.24	0.56/0.3	0.45/0.3	0.67/0.24
	DT	0.78/0.01	0.63/0.09	0.47/0.14	0.51/0.11	0.5/0.12	0.58/0.12
	RF	0.81/0.01	0.69/0.11	0.47/0.14	0.52/0.14	0.52/0.15	0.6/0.14
	MLP	0.68/0.08	0.53/0.17	0.49/0.06	0.48/0.03	0.47/0.17	0.57/0.06
4*TVAE	LR	0.76/0.02	0.77/0.01	0.72/0.16	0.38/0.16	0.45/0.28	0.48/0.22
	DT	0.77/0.01	0.77/0.03	0.61/0.09	0.47/0.04	0.48/0.11	0.52/0.06
	RF	0.81/0.01	0.8/0.01	0.66/0.13	0.43/0.05	0.47/0.16	0.53/0.08
	MLP	0.67/0.05	0.75/0.04	0.53/0.12	0.53/0.08	0.5/0.16	0.51/0.04
4*CopulaGAN	LR	0.78/0.02	0.77/0.0	0.45/0.3	0.65/0.25	0.55/0.3	0.46/0.3
	DT	0.76/0.01	0.72/0.03	0.51/0.16	0.55/0.13	0.55/0.14	0.5/0.11
	RF	0.81/0.01	0.77/0.02	0.52/0.2	0.56/0.13	0.56/0.15	0.5/0.13
	MLP	0.63/0.08	0.6/0.08	0.48/0.12	0.54/0.15	0.51/0.1	0.52/0.06

Table 6.2: Machine learning efficacy with two different noisy label generation methods. The mean and standard deviation values of accuracy are reported in each cell.

6.5.3. Impacts on downstream classification tasks. Empirical results on impacts on downstream tasks are presented in Figure 5 and Table 6.2. We observed this aspect using two different metrics: machine learning efficacy and learning curve in training. We observe that machine learning efficacy in terms of accuracy decreases as higher noise levels are used in generating synthetic data as shown in Figure 5. Detailed results with standard deviation are displayed in Supplementary Table 6.2. There are some interesting situations 1) both accuracy and noise level increase together, e.g., level 30% - 50% in TVAE for an instance-dependent noise generation system, and 2) high variance in performances (Table 6.2). High variances are observed in both random flipping and instance-dependent noises similarly and results from random flipping show relatively more consistent behavior than instance-dependent noise generation. High variances can be partially explained by fluctuations possibly caused by 1) noise label generation, 2) instability of training synthesizer, and 3) training machine learning models with synthetic data, as each situation has chances to affect the randomness when we work with label noise in the beginning. We also observe that performance gaps between different machine learning methods are not very significant. We also observed the training/validation analysis in machine learning downstream models using learning curves. It is observed from learning curves on training data (synthetic data) and validation data (real data) have accuracy gaps, i.e., overfitting and convergence delay. Also, even though it is not very significant, larger noises outputted larger gaps in accuracy as shown in Figure 6. Additionally, Sometimes learning curves on training data with larger noise levels show better performance than those with smaller noise levels, e.g., CopulaGAN with random flipping noise type but performance behaviors become coherent with the noise levels in the learning curve on validation data. Such behaviors are observed more in the instance-dependent label noise approach.

6.6. Discussion

We provide the first comprehensive study to investigate the effects of label noise when generating a synthetic tabular dataset. To end this, we adopt four different aspects: synthesizer convergence rate, data quality (statistical and sample-level similarity, label accuracy) data utility (machine learning efficacy). Adult data that consists of 32k samples with 14 attributes including the label is utilized for experiments including downstream binary classification tasks, i.e., annual income level; over 50k or not. Our extensive numerical analysis of the impact of label noise on synthetic tabular data generation provides valuable insights into the understanding of label noise. By examining the effects of label noise on various aspects of data generation, researchers and practitioners can gain potential mitigating strategies associated with noisy labels. Furthermore, this analysis can serve as a foundation for the development of more robust models and techniques for dealing with label noise in synthetic data generation, ultimately improving the quality and reliability of machine learning applications.

In our experiments, training the synthesizer component involves two phases: pre-training with real data and fine-tuning with noisy data. This is crucial for stabilizing deep generative-based synthesizers, as without pre-training, the synthesizer may become unstable and impact all subsequent components, leading to inconclusive results regarding the sole effect of label noise on synthetic tabular data generation.

Despite our best efforts to isolate the impact of label noise, our study's limitation lies in the various factors that can affect the metrics. Even though we employ pre-training synthesizers to enhance their stability, there is a still chance that the metrics for generated synthetic data can be affected by intermediate steps such as noise generation, the instability of the training synthesizer, and training downstream machine learning models with synthetic data. Regarding this issue, we employ label accuracy as a metric because we observe high variances in machine efficacy measurements, which can possibly be caused by a chance that noisy data influenced machine learning training for downstream classification tasks. Therefore, identifying the specific impact of label noise in each step, such as training synthesizers and downstream tasks within the entire framework, remains a challenging task and an important next step.

Numerous studies have emphasized the privacy concerns related to synthetic data generation [1, 2, 3]. In order to further improve privacy in synthetic data generation, it is significant to examine the influence of label noise on privacy aspects [**13**, **36**, **39**]. Conducting such investigations helps us develop more robust strategies to address privacy concerns in synthetic data generation methodologies. By gaining a deeper understanding of the relationship between label noise and privacy, we can contribute to the development of secure and reliable synthetic data generation techniques, ultimately benefiting the broader research community and the applications that rely on these methods.

6.7. Conclusion

In this study, we explore the influence of label noise on synthetic tabular data generation. Our primary objectives are to evaluate the quality and utility of the synthetic tabular data and its associated labels, as well as to estimate the impact on synthesizer convergence. To this end, we conduct a series of comprehensive experiments that assess the performance of the synthetic data generation process under varying levels of label noise.

Our findings demonstrate that label noise significantly undermines the quality and utility of the generated tabular data, as well as the accuracy of the synthetic labels. Interestingly, we also observe that synthesizer convergence is not substantially affected by label noise. These results highlight the complex relationship between label noise and data generation, particularly in the context of tabular data. By examining these nuanced effects, researchers can gain valuable insight into the factors that influence the quality and reliability of synthetic tabular data generated amidst label noise.

Through our examination of the influence of label noise on synthetic tabular data generation, this study contributes to a more profound understanding of the challenges and limitations inherent in creating high-quality data in the presence of label noise. The knowledge derived from this research can potentially inform the development of more effective denoising strategies and algorithms, ultimately enhancing the privacy and utility of synthetic data in real-world applications.

Random Flipping Label Noise

2*Synthesizers	2*ML models	Noise Level					
		0.0	0.1	0.2	0.3	0.4	0.5
4*CTGAN	LR	0.78/0.0	0.72/0.0	0.71/0.0	0.43/0.0	0.5/0.0	0.41/0.0
	DT	0.79/0.01	0.72/0.0	0.66/0.01	0.46/0.0	0.51/0.0	0.45/0.0
	RF	0.83/0.0	0.75/0.0	0.68/0.0	0.45/0.0	0.51/0.0	0.43/0.0
	MLP	0.56/0.3	0.55/0.23	0.71/0.01	0.49/0.1	0.5/0.0	0.49/0.11
4*TVAE	LR	0.77/0.0	0.79/0.0	0.78/0.0	0.78/0.0	0.52/0.0	0.7/0.0
	DT	0.83/0.0	0.84/0.0	0.8/0.01	0.78/0.01	0.57/0.01	0.61/0.01
	RF	0.85/0.01	0.86/0.0	0.82/0.0	0.8/0.01	0.57/0.01	0.63/0.0
	MLP	0.69/0.16	0.67/0.25	0.67/0.25	0.67/0.25	0.52/0.0	0.69/0.02
4*CopulaGAN	LR	0.82/0.0	0.78/0.0	0.47/0.0	0.6/0.0	0.52/0.0	0.46/0.0
	DT	0.79/0.0	0.74/0.0	0.49/0.0	0.58/0.0	0.52/0.0	0.47/0.0
	RF	0.85/0.0	0.78/0.0	0.49/0.0	0.6/0.0	0.52/0.0	0.47/0.0
	MLP	0.69/0.29	0.66/0.25	0.49/0.03	0.53/0.11	0.52/0.01	0.46/0.0

Instance-dependent Label Noise

2*Synthesizers	2*ML models	Noise Level					
		0.0	0.1	0.2	0.3	0.4	0.5
4*CTGAN	LR	0.72/0.0	0.73/0.0	0.62/0.0	0.56/0.0	0.48/0.0	0.6/0.0
	DT	0.75/0.0	0.73/0.0	0.63/0.0	0.55/0.0	0.5/0.0	0.56/0.0
	RF	0.79/0.0	0.76/0.0	0.64/0.0	0.56/0.0	0.49/0.0	0.56/0.0
	MLP	0.72/0.0	0.61/0.2	0.62/0.0	0.51/0.06	0.49/0.02	0.52/0.1
4*TVAE	LR	0.81/0.0	0.82/0.0	0.79/0.0	0.84/0.0	0.55/0.0	0.27/0.0
	DT	0.82/0.0	0.81/0.0	0.8/0.0	0.8/0.0	0.59/0.01	0.32/0.01
	RF	0.85/0.0	0.83/0.0	0.83/0.0	0.82/0.0	0.6/0.0	0.31/0.0
	MLP	0.69/0.28	0.56/0.36	0.79/0.01	0.84/0.0	0.53/0.05	0.36/0.21
4*CopulaGAN	LR	0.76/0.0	0.83/0.0	0.71/0.0	0.68/0.0	0.37/0.0	0.4/0.0
	DT	0.77/0.0	0.78/0.0	0.67/0.0	0.65/0.01	0.41/0.0	0.45/0.0
	RF	0.82/0.0	0.81/0.0	0.7/0.0	0.66/0.0	0.4/0.0	0.44/0.0
	MLP	0.76/0.0	0.83/0.0	0.63/0.19	0.53/0.19	0.37/0.0	0.44/0.09

Table 6.3: Label accuracy with two different noisy label generation methods. The mean and standard deviation values of accuracy are reported in each cell.

Random Flipping Label Noise

2*Synthesizers	2*Metrics	Noise Level				
		0.1	0.2	0.3	0.4	0.5
3*CTGAN	α-precision	1.02/0.08	0.39/0.98	0.34/1.02	-0.9/0.01	0.37/1.11
	β-recall	1.06/0.01	0.82/0.21	0.81/0.22	0.68/0.04	0.84/0.23
	Authenticity	0.0/0.0	0.24/0.38	0.24/0.37	0.69/0.01	0.24/0.39
3*TVAE	α-precision	0.58/0.85	0.51/0.85	0.57/0.87	0.02/0.91	0.64/0.61
	β-recall	0.87/0.34	0.88/0.31	0.87/0.35	0.68/0.33	0.54/0.65
	Authenticity	0.23/0.39	0.24/0.42	0.24/0.41	0.46/0.4	0.15/0.19
3*CopulaGAN	α-precision	0.32/1.03	0.35/1.09	0.3/1.07	0.32/0.97	0.99/0.12
	β-recall	0.82/0.37	0.77/0.5	0.73/0.33	0.82/0.37	1.05/0.02
	Authenticity	0.22/0.36	0.19/0.33	0.24/0.36	0.22/0.36	0.0/0.0

Instance-dependent Label Noise

2*Synthesizers	2*Metrics	Noise Level				
		0.1	0.2	0.3	0.4	0.5
3*CTGAN	α-precision	0.98/0.08	1.02/0.08	0.46/0.99	1.02/0.08	-0.15/0.93
	β-recall	1.02/0.04	1.06/0.01	0.73/0.36	1.02/0.05	0.57/0.4
	Authenticity	0.01/0.01	0.0/0.0	0.23/0.36	0.01/0.01	0.43/0.36
3*TVAE	α-precision	1.02/0.08	0.55/0.9	1.07/0.0	0.26/0.7	0.64/0.63
	β-recall	1.05/0.02	0.81/0.43	1.06/0.02	0.59/0.38	0.84/0.39
	Authenticity	0.01/0.0	0.22/0.38	0.0/0.0	0.44/0.38	0.22/0.38
3*CopulaGAN	α-precision	0.39/1.06	0.35/0.99	0.97/0.08	-0.3/1.07	-0.81/0.03
	β-recall	0.82/0.39	0.74/0.34	0.9/0.21	0.57/0.41	0.31/0.0
	Authenticity	0.22/0.36	0.23/0.36	0.03/0.04	0.42/0.35	0.63/0.01

Table 6.4: Sample-Level comparison with two different noisy label generation methods. The mean and standard deviation values of accuracy are reported in each cell.

Bibliography

[1] A. M. Alaa, B. van Breugel, E. Saveliev, and M. van der Schaar, *How Faithful is your Synthetic Data? Sample-level Metrics for Evaluating and Auditing Generative Models*, (2021).

[2] M. A. Aldeyab, D. L. Monnet, J. M. López-Lozano, C. M. Hughes, M. G. Scott, M. P. Kearney, F. A. Magee, and J. C. McElnay, *Modelling the impact of antibiotic use and infection control practices on the incidence of hospital-acquired methicillin-resistant Staphylococcus aureus: a time-series analysis*, Journal of Antimicrobial Chemotherapy, 62 (2008), pp. 593–600.

[3] J. An and S. Cho, *Variational autoencoder based anomaly detection using reconstruction probability*, Special Lecture on IE, 2 (2015), pp. 1–18.

[4] W. R. Atchley, J. Zhao, A. D. Fernandes, and T. Drüke, *Solving the protein sequence metric problem*, Proceedings of the National Academy of Sciences of the United States of America, 102 (2005), pp. 6395–6400.

[5] M. D. Barton, *Impact of antibiotic use in the swine industry*, Current opinion in microbiology, 19 (2014), pp. 9–15.

[6] A. R. Benaim, R. Almog, Y. Gorelik, I. Hochberg, L. Nassar, T. Mashiach, M. Khamaisi, Y. Lurie, Z. S. Azzam, J. Khoury, et al., *Analyzing medical research results based on synthetic data and their relation to real data results: systematic comparison from five observational studies*, JMIR medical informatics, 8 (2020), p. e16492.

[7] M. Benjamin and S. Yik, *Precision Livestock Farming in Swine Welfare: A Review for Swine Practitioners*, 2019.

[8] V. Borisov, K. Sessler, T. Leemann, M. Pawelczyk, and G. Kasneci, *Language models are realistic tabular data generators*, (2023).

[9] L. Breiman, *Random forests*, 2001.

[10] E. J. Candès, X. Li, Y. Ma, and J. Wright, *Robust principal component analysis*, Journal of the ACM (JACM), 58 (2011), pp. 1–37.

[11] S. H. Cha, C. C. Chang, and K. J. Yoon, *Instability of the restriction fragment length polymorphism pattern of open reading frame 5 of porcine reproductive and respiratory syndrome virus during sequential pig-to-pig passages*, Journal of Clinical Microbiology, 42 (2004), pp. 4462–4467.

[12] C. Chatfield, *The analysis of time series: an introduction*, Chapman and hall/CRC, 2003.

[13] D. Chen, N. Yu, Y. Zhang, and M. Fritz, *GAN-Leaks: A Taxonomy of Membership Inference Attacks against Generative Models*, Proceedings of the ACM Conference on Computer and Communications Security, (2020), pp. 343–362.

[14] R. J. Chen, M. Y. Lu, T. Y. Chen, D. F. Williamson, and F. Mahmood, *Synthetic data in machine learning for medicine and healthcare*, Nature Biomedical Engineering, 5 (2021), pp. 493–497.

[15] I. Correas, F. A. Osorio, D. Steffen, A. K. Pattnaik, and H. L. Vu, *Cross reactivity of immune responses to porcine reproductive and respiratory syndrome virus infection*, Vaccine, 35 (2017), pp. 782–788.

[16] C. Cortes and V. Vapnik, *Support-Vector Networks*, Machine Learning, 20 (1995), pp. 273–297.

[17] T. M. Cover and P. E. Hart, *Nearest Neighbor Pattern Classification*, IEEE Transactions on Information Theory, 13 (1967), pp. 21–27.

[18] T. M. Cover and J. A. Thomas, *Elements of Information Theory*, 2005.

[19] P. Cwynar, J. Stojkov, and K. Wlazlak, *African swine fever status in europe*, Viruses, 11 (2019), p. 310.

[20] J. Dahmen and D. Cook, *Synsys: A synthetic data generation system for healthcare applications*, Sensors, 19 (2019), p. 1181.

[21] J. Dewulf, B. Catry, T. Timmerman, G. Opsomer, A. de Kruif, and D. Maes, *Tetracycline-resistance in lactose-positive enteric coliforms originating from Belgian fattening pigs: Degree of resistance, multiple resistance and risk factors*, Preventive Veterinary Medicine, 78 (2007), pp. 339–351.

[22] H. I. Dino and M. B. Abdulrazzaq, *Facial Expression Classification Based on SVM, KNN and MLP Classifiers*, 2019 International Conference on Advanced Science and Engineering, ICOASE 2019, (2019), pp. 70–75.

[23] C. Dong, L. Liu, and J. Shang, *Label Noise in Adversarial Training: A Novel Perspective to Study Robust Overfitting*, in Advances in Neural Information Processing Systems, S. Koyejo, S. Mohamed, A. Agarwal, D. Belgrave, K. Cho, and A. Oh, eds., vol. 35, Curran Associates, Inc., 2022, pp. 17556–17567.

[24] D. Dua and C. Graff, *UCI machine learning repository*, 2017.

[25] M. Erbert, S. Rechner, and M. Müller-Hannemann, *Gerbil: a fast and memory-efficient k-mer counter with GPU-support*, Algorithms for Molecular Biology, 12 (2017), p. 9.

[26] S. M. Fazel, *Matrix rank minimization with applications.*, (2003).

[27] FDA, U.S Department of Health and Human Services, Food and Drug Administration, and Center for Devices and Radiological Health, *Guidance for Industry and FDA. Class II Special Controls Guidance Document: Antimicrobial Susceptibility Test (AST) Systems*, (2009), pp. 1–42.

[28] FDA Department of Health and Human Services, *Summary Report on Antimicrobials Sold or Distributed for Use in Food-Producing Animals.*, (2011).

[29] B. Goldluecke, *Total Variation BT - Computer Vision: A Reference Guide*, Springer International Publishing, Cham, 2021, pp. 1266–1269.

[30] I. J. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio, *Generative adversarial networks*, 2014.

[31] I. J. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio, *Generative adversarial networks*, (2014).

[32] K. Gu, X. Masotto, V. Bachani, B. Lakshminarayanan, J. Nikodem, and D. Yin, *An instance-dependent simulation framework for learning with label noise*, (2021).

[33] W. Guo, F. Sun, F. Liu, L. Cao, J. Yang, and Y. Chen, *Antimicrobial resistance surveillance and prediction of Gram-negative bacteria based on antimicrobial consumption in a hospital setting: A 15-year retrospective study*, Medicine, 98 (2019).

[34] Z. Guo, B. Yu, M. Hao, W. Wang, Y. Jiang, and F. Zong, *A novel hybrid method for flight departure delay prediction using Random Forest Regression and Maximal Information Coefficient*, Aerospace Science and Technology, 116 (2021), p. 106822.

[35] J. A. Hanley and B. J. McNeil, *The meaning and use of the area under a receiver operating characteristic (ROC) curve.*, Radiology, 143 (1982), pp. 29–36.

[36] J. Hayes, L. Melis, G. Danezis, and E. De Cristofaro, *LOGAN: Membership Inference Attacks Against Generative Models*, Proceedings on Privacy Enhancing Technologies, 2019 (2019), pp. 133–152.

[37] D. Hendrycks, K. Lee, and M. Mazeika, *Using pre-training can improve model robustness and uncertainty*, (2019).

[38] M. Hernandez, G. Epelde, A. Alberdi, R. Cilla, and D. Rankin, *Synthetic data generation for tabular health records: A systematic review*, Neurocomputing, (2022).

[39] B. Hitaj, G. Ateniese, and F. Perez-Cruz, *Deep Models under the GAN: Information leakage from collaborative deep learning*, Proceedings of the ACM Conference on Computer and Communications Security, 1 (2017), pp. 603–618.

[40] D. J. Holtkamp, J. B. Kliebenstein, E. J. Neumann, J. J. Zimmerman, H. F. Rotto, T. K. Yoder, C. Wang, P. E. Yeske, C. L. Mowrer, and C. A. Haley, *Assessment of the economic impact of porcine reproductive and respiratory syndrome virus on United States pork producers*, Journal of Swine Health and Production, 21 (2013), pp. 72–84.

[41] B. Howe, J. Stoyanovich, H. Ping, B. Herman, and M. Gee, *Synthetic data for social good*, arXiv preprint arXiv:1710.08874, (2017).

[42] P.-R. Hsueh, W.-H. Chen, and K.-T. Luh, *Relationships between antimicrobial use and antimicrobial resistance in Gram-negative bacteria causing nosocomial infections from 1991–2003 at a university hospital in Taiwan*, International Journal of Antimicrobial Agents, 26 (2005), pp. 463–472.

[43] K. J. In, M. Finlay, T. K. K., G. Theodore, P. S. J., M. T. A., M. A. G., and B. R. G., *Machine Learning for Antimicrobial Resistance Prediction: Current Practice, Limitations, and Clinical Perspective*, Clinical Microbiology Reviews, 35 (2022), pp. e00179–21.

[44] B. Jeffrey, D. M. Aanensen, N. J. Croucher, and S. Bhatt, *Predicting the future distribution of antibiotic resistance using time series forecasting and geospatial modelling*, Wellcome Open Research, 5 (2021), pp. 1–26.

[45] J. M. Johnson and T. M. Khoshgoftaar, *A survey on classifying big data with label noise*, J. Data and Information Quality, 14 (2022).

[46] S. Kalyaanamoorthy, B. Q. Minh, T. K. Wong, A. Von Haeseler, and L. S. Jermiin, *ModelFinder: Fast model selection for accurate phylogenetic estimates*, Nature Methods, 14 (2017), pp. 587–589.

[47] S. Kamthe, S. Assefa, and M. Deisenroth, *Copula flows for synthetic data generation*, (2021).

[48] T. Kaneko, Y. Ushiku, and T. Harada, *Label-noise robust generative adversarial networks*, Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition, 2019-June (2019), pp. 2462–2471.

[49] ———, *Label-noise robust generative adversarial networks*, Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition, 2019-June (2019), pp. 2462–2471.

[50] S. Ke, C. Huang, and X. Liu, *Quantifying the impact of label noise on federated learning*, 2023.

[51] H.-J. Kim, K.-H. Cho, S.-K. Lee, D.-Y. Kim, J.-J. Nah, H.-J. Kim, H.-J. Kim, J.-Y. Hwang, H.-J. Sohn, J.-G. Choi, et al., *Outbreak of african swine fever in south korea, 2019*, Transboundary and emerging diseases, 67 (2020), pp. 473–475.

[52] K. H. Kim, S. Shim, Y. Lim, J. Jeon, J. Choi, B. Kim, and A. S. Yoon, *Rapp: Novelty detection with reconstruction along projection pathway*, in International Conference on Learning Representations, 2019.

[53] W. I. Kim, J. J. Kim, S. H. Cha, W. H. Wu, V. Cooper, R. Evans, E. J. Choi, and K. J. Yoon, *Significance of genetic variation of PRRSV ORF5 in virus neutralization and molecular determinants corresponding to cross neutralization among PRRS viruses*, Veterinary Microbiology, 162 (2013), pp. 10–22.

[54] N. Kodali, J. Abernethy, J. Hays, and Z. Kira, *On convergence and stability of gans*, (2017).

[55] M. Kubat, *Neural networks: a comprehensive foundation by Simon Haykin, Macmillan, 1994, ISBN 0-02-352781-7.* , 1999.

[56] B. Kwon, I. H. Ansari, A. K. Pattnaik, and F. A. Osorio, *Identification of virulence determinants of porcine reproductive and respiratory syndrome virus through construction of chimeric clones*, Virology, 380 (2008), pp. 371–378.

[57] A. Larsson, *AliView: A fast and lightweight alignment viewer and editor for large datasets*, Bioinformatics, 30 (2014), pp. 3276–3278.

[58] G. Lederrey, T. Hillel, and M. Bierlaire, *Datgan: Integrating expert knowledge into deep learning for synthetic tabular data*, 2022.

[59] W. Lee, Y. Ham, T.-W. Ban, and O. Jo, *Analysis of growth performance in swine based on machine learning*, IEEE Access, 7 (2019), pp. 161716–161724.

[60] I. Letunic and P. Bork, *Interactive Tree of Life (iTOL) v4: Recent updates and new developments*, Nucleic Acids Research, 47 (2019), pp. 256–259.

[61] J. Lezama, Q. Qiu, P. Musé, and G. Sapiro, *Ole: Orthogonal low-rank embedding-a plug and play geometric loss for deep learning*, in Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2018, pp. 8109–8118.

[62] R. Liang, Y. Lu, X. Qu, Q. Su, C. Li, S. Xia, Y. Liu, Q. Zhang, X. Cao, Q. Chen, et al., *Prediction for global african swine fever outbreaks based on a combination of random forest algorithms and meteorological data*, Transboundary and emerging diseases, 67 (2020), pp. 935–946.

[63] T. Liu, Z. Qian, J. Berrevoets, and M. van der Schaar, *Goggle: Generative modelling for tabular data by learning relational structure*, in International Conference on Learning Representations, 2023.

[64] J.-M. López-Lozano, D. L. Monnet, A. Yagüe, A. Burgos, N. Gonzalo, P. Campillos, and M. Saez, *Modelling and forecasting antimicrobial resistance and its dynamic relationship to antimicrobial use: a time series analysis*, International Journal of Antimicrobial Agents, 14 (2000), pp. 21–31.

[65] G. Louppe, *Understanding Random Forests: From Theory to Practice*, PhD thesis, 10 2014.

[66] B. V. Lubbers, D. Diaz-Campos, S. Schwarz, and Clinical and Laboratory Standards Institute, *Performance standards for antimicrobial disk and dilution susceptibility tests for bacteria isolated from animals*, (2020), p. 216.

[67] R. E. C. Luiken, D. J. J. Heederik, P. Scherpenisse, L. Van Gompel, E. van Heijnsbergen, G. D. Greve, B. G. M. Jongerius-Gortemaker, M. H. G. Tersteeg-Zijderveld, J. Fischer, K. Juraschek, M. Skarżyńska, M. Zając, D. Wasyl, J. A. Wagenaar, L. A. M. Smit, I. M. Wouters, D. J. Mevius, and H. Schmitt, *Determinants for antimicrobial resistance genes in farm dust on 333 poultry and pig farms in nine European countries*, Environmental Research, 208 (2022), p. 112715.

[68] S. Ma, Q. Liu, and Y. Zhang, *A prediction method of fire frequency: Based on the optimization of SARIMA model*, PLoS ONE, 16 (2021), pp. 1–13.

[69] W. Ma, *Swine influenza virus: Current status and challenge*, Virus Research, 288 (2020), p. 198118.

[70] X. Ma, H. Huang, Y. Wang, S. Romano, S. Erfani, and J. Bailey, *Normalized loss functions for deep learning with noisy labels*, in Proceedings of the 37th International Conference on Machine Learning, H. D. III and A. Singh, eds., vol. 119 of Proceedings of Machine Learning Research, PMLR, 13–18 Jul 2020, pp. 6543–6553.

[71] A. Makhzani, J. Shlens, N. Jaitly, I. Goodfellow, and B. Frey, *Adversarial autoencoders*, arXiv preprint arXiv:1511.05644, (2015).

[72] N. Marcus, L. S. Wesley, M. P. F., O. R. J., O. Robert, S. R. L., T. G. H., Z. Shaohua, and D. J. J., *Using Machine Learning To Predict Antimicrobial MICs and Associated Genomic Features for Nontyphoidal Salmonella*, Journal of Clinical Microbiology, 57 (2019), pp. e01260–18.

[73] D. Mason-D'Croz, J. R. Bogard, M. Herrero, S. Robinson, T. B. Sulser, K. Wiebe, D. Willenbockel, and H. C. J. Godfray, *Modelling the global economic consequences of a major african swine fever outbreak in china*, Nature Food, 1 (2020), pp. 221–228.

[74] A. G. Mathew, D. B. Arnett, P. Cullen, and P. D. Ebner, *Characterization of resistance patterns and detection of apramycin resistance genes in Escherichia coli isolated from swine exposed to various environmental conditions*, International Journal of Food Microbiology, 89 (2003), pp. 11–20.

[75] J. J. Medardus, B. Z. Molla, M. Nicol, W. M. Morrow, P. J. Rajala-Schultz, R. Kazwala, and W. A. Gebreyes, *In-feed use of heavy metal micronutrients in U.S. swine production systems and its role in persistence of multidrug-resistant salmonellae*, Applied and Environmental Microbiology, 80 (2014), pp. 2317–2325.

[76] F. Murtagh, *Multilayer perceptrons for classification and regression*, Neurocomputing, 2 (1991), pp. 183–197.

[77] A. M. N., Y. J. H., and K. Sanjat, *Applications of Machine Learning to the Problem of Antimicrobial Resistance: an Emerging Model for Translational Research*, Journal of Clinical Microbiology, 59 (2021), pp. e01260–20.

[78] Y. Nan, C. Wu, G. Gu, W. Sun, Y. J. Zhang, and E. M. Zhou, *Improved vaccine against PRRSV: Current Progress and future perspective*, Frontiers in Microbiology, 8 (2017), pp. 1–17.

[79] Narms, *National Antimicrobial Resistance Monitoring System 2011 Executive Report*, Health education research, 29 (2011).

[80] L. T. Nguyen, H. A. Schmidt, A. Von Haeseler, and B. Q. Minh, *IQ-TREE: A fast and effective stochastic algorithm for estimating maximum-likelihood phylogenies*, Molecular Biology and Evolution, 32 (2015), pp. 268–274.

[81] M. Nguyen, T. Brettin, S. W. Long, J. M. Musser, R. J. Olsen, R. Olson, M. Shukla, R. L. Stevens, F. Xia, H. Yoo, and J. J. Davis, *Developing an in silico minimum inhibitory concentration panel test for Klebsiella pneumoniae*, Scientific Reports, 8 (2018), p. 421.

[82] M. Nguyen, S. W. Long, P. F. McDermott, R. J. Olsen, R. Olson, R. L. Stevens, G. H. Tyson, S. Zhao, and J. J. Davis, *Using Machine Learning To Predict Antimicrobial MICs and Associated Genomic Features for Nontyphoidal Salmonella*, Journal of Clinical Microbiology, 57 (2019), pp. e01260–18.

[83] M. Nguyen, S. Wesley Long, P. F. McDermott, R. J. Olsen, R. Olson, R. L. Stevens, G. H. Tyson, S. Zhao, and J. J. Davisa, *Using machine learning to predict antimicrobial MICs and associated genomic features for nontyphoidal Salmonella*, Journal of Clinical Microbiology, 57 (2019).

[84] M. Ostrowski, J. A. Galeota, A. M. Jar, K. B. Platt, F. A. Osorio, and O. J. Lopez, *Identification of Neutralizing and Nonneutralizing Epitopes in the Porcine Reproductive and Respiratory Syndrome Virus GP5 Ectodomain*, Journal of Virology, 76 (2002), pp. 4241–4250.

[85] A. Papacharalampopoulos, K. Tzimanis, K. Sabatakakis, and P. Stavropoulos, *Deep quality assessment of a solar reflector based on synthetic data: detecting surficial defects from manufacturing and use phase*, Sensors, 20 (2020), p. 5481.

[86] I. A. D. Paploski, C. Corzo, A. Rovira, M. P. Murtaugh, J. M. Sanhueza, C. Vilalta, D. C. Schroeder, and K. VanderWaal, *Temporal Dynamics of Co-circulating Lineages of Porcine Reproductive and Respiratory Syndrome Virus*, Frontiers in Microbiology, 10 (2019), pp. 1–13.

[87] M. Park, *Jgan: A joint formulation of gan for synthesizing images and labels*, IEEE Access, 8 (2020), pp. 188883–188888.

[88] K. Pearson, *X. on the criterion that a given system of deviations from the probable in the case of a correlated system of variables is such that it can be reasonably supposed to have arisen from random sampling*, The London, Edinburgh, and Dublin Philosophical Magazine and Journal of Science, 50 (1900), pp. 157–175.

[89] P. Perera, R. Nallapati, and B. Xiang, *Ocgan: One-class novelty detection using gans with constrained latent representations*, in Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2019, pp. 2898–2906.

[90] M. W. Pesesky, T. Hussain, M. Wallace, S. Patel, S. Andleeb, C.-A. D. Burnham, and G. Dantas, *Evaluation of Machine Learning and Rules-Based Approaches for Predicting Antimicrobial Resistance Profiles in Gram-negative Bacilli from Whole Genome Sequence Data*, Frontiers in Microbiology, 7 (2016).

[91] S. Pidhorskyi, R. Almohsen, and G. Doretto, *Generative probabilistic novelty detection with adversarial autoencoders*, in Advances in neural information processing systems, 2018, pp. 6822–6833.

[92] X. Qiang, Z. Kou, G. Fang, and Y. Wang, *Scoring Amino Acid Mutations to Predict Avian-to-Human Transmission of Avian Influenza Viruses.*, Molecules (Basel, Switzerland), 23 (2018).

[93] Q. Qiu and G. Sapiro, *Learning transformations for clustering and classification*, The Journal of Machine Learning Research, 16 (2015), pp. 187–225.

[94] S. Rashidian, F. Wang, R. Moffitt, V. Garcia, A. Dutt, W. Chang, V. Pandya, J. Hajagos, M. Saltz, and J. Saltz, *SMOOTH-GAN: Towards Sharp and Smooth Synthetic EHR Data Generation BT - Artificial Intelligence in Medicine*, Cham, 2020, Springer International Publishing, pp. 37–48.

[95] B. Recht, M. Fazel, and P. A. Parrilo, *Guaranteed minimum-rank solutions of linear matrix equations via nuclear norm minimization*, SIAM review, 52 (2010), pp. 471–501.

[96] A. Reza Hoshmand, *Business Forecasting: A Practical Approach*, 2009.

[97] G. Rizk, D. Lavenier, and R. Chikhi, *DSK: k-mer counting with very low memory usage*, Bioinformatics, 29 (2013), pp. 652–653.

[98] R. C. Robbins, G. Almond, and E. Byers, *Swine Diseases and Disorders*, Academic Press, Oxford, 2014, pp. 261–276.

[99] K. Ryu and A. Sanchez, *The evaluation of forecasting methods at an institutional foodservice dining facility*, Journal of Hospitality Financial Management, 11 (2003), pp. 27–45.

[100] O. Räisä, J. Jälkö, S. Kaski, and A. Honkela, *Noise-aware statistical inference with differentially private synthetic data*, 2023.

[101] P. Sagulenko, V. Puller, and R. A. Neher, *TreeTime: Maximum-likelihood phylodynamic analysis*, Virus Evolution, 4 (2018), p. vex042.

[102] M. Sakurada and T. Yairi, *Anomaly detection using autoencoders with nonlinear dimensionality reduction*, in Proceedings of the MLSDA 2014 2nd Workshop on Machine Learning for Sensory Data Analysis, 2014, pp. 4–11.

[103] T. Schlegl, P. Seeböck, S. M. Waldstein, U. Schmidt-Erfurth, and G. Langs, *Unsupervised anomaly detection with generative adversarial networks to guide marker discovery*, in International conference on information processing in medical imaging, Springer, 2017, pp. 146–157.

[104] M. Shamsabardeh, S. Rezaei, J. P. Gomez, B. Martínez-Lóṕez, and X. Liu, *A novel way to predict prrs outbreaks in the swine industry using multiple spatio-temporal features and machine learning approaches*, Frontiers in Veterinary Science, 6 (2019).

[105] M. Shi, T. T. Y. Lam, C. C. Hon, R. K. H. Hui, K. S. Faaberg, T. Wennblom, M. P. Murtaugh, T. Stadejek, and F. C. C. Leung, *Molecular epidemiology of PRRSV: A phylogenetic perspective*, Virus Research, 154 (2010), pp. 7–17.

[106] M. Shi, T. T.-Y. Lam, C.-C. Hon, M. P. Murtaugh, P. R. Davies, R. K.-H. Hui, J. Li, L. T.-W. Wong, C.-W. Yip, J.-W. Jiang, and F. C.-C. Leung, *Phylogeny-Based Evolutionary, Demographical, and Geographical Dissection of North American Type 2 Porcine Reproductive and Respiratory Syndrome Viruses*, Journal of Virology, 84 (2010), pp. 8700–8711.

[107] P. P. Shinde and S. Shah, *A Review of Machine Learning and Deep Learning Applications*, Proceedings - 2018 4th International Conference on Computing, Communication Control and Automation, ICCUBEA 2018, (2018), pp. 1–6.

[108] A. Statnikov, L. Wang, and C. F. Aliferis, *A comprehensive comparison of random forests and support vector machines for microarray-based cancer classification*, BMC Bioinformatics, 9 (2008), pp. 1–10.

[109] D. Strahlberg, *Antibiotics Resistance Forecasting: A Comparison of Two Time Series Forecast Models*, SIAM Undergraduate Research Online, 14 (2021), pp. 383–399.

[110] R. Tan, A. Yu, Z. Liu, Z. Liu, R. Jiang, X. Wang, J. Liu, J. Gao, and X. Wang, *Prediction of Minimal Inhibitory Concentration of Meropenem Against Klebsiella pneumoniae Using Metagenomic Data*, Frontiers in Microbiology, 12 (2021), pp. 1–10.

[111] B. Thaa, B. C. Sinhadri, C. Tielesch, E. Krause, and M. Veit, *Signal Peptide Cleavage from GP5 of PRRSV: A Minor Fraction of Molecules Retains the Decoy Epitope, a Presumed Molecular Cause for Viral Persistence*, PLoS ONE, 8 (2013).

[112] K. K. Thekumparampil, A. Khetan, Z. Lin, and S. Oh, *Robustness of conditional GANs to noisy labels*, Advances in Neural Information Processing Systems, 2018-December (2018), pp. 10271–10282.

[113] ———, *Robustness of conditional GANs to noisy labels*, Advances in Neural Information Processing Systems, 2018-December (2018), pp. 10271–10282.

[114] K. Ting, *Particle swarm optimisation*, Studies in Computational Intelligence, 780 (2019), pp. 15–31.

[115] M. Valipour, M. E. Banihabib, and S. M. R. Behbahani, *Parameters estimate of autoregressive moving average and autoregressive integrated moving average models and compare their ability for inflow forecasting*, Journal of Mathematics and Statistics, 8 (2012), pp. 330–338.

[116] T. van Klompenburg and A. Kassahun, *Data-driven decision making in pig farming: A review of the literature*, Livestock Science, 261 (2022), p. 104961.

[117] A. Wagenmaker and K. G. Jamieson, *Instance-dependent near-optimal policy identification in linear mdps via online experiment design*, Advances in Neural Information Processing Systems, 35 (2022), pp. 5968–5981.

[118] H. Wang, C. Jia, H. Li, R. Yin, J. Chen, Y. Li, and M. Yue, *Paving the way for precise diagnostics of antimicrobial resistant bacteria*, Frontiers in Molecular Biosciences, 9 (2022).

[119] J. L. Watts, M. T. Sweeney, and B. V. Lubbers, *Antimicrobial Susceptibility Testing of Bacteria of Veterinary Origin*, Microbiology Spectrum, 6 (2018).

[120] R. D. Wesley, W. L. Mengeling, K. M. Lager, D. F. Clouser, J. G. Landgraf, and M. L. Frey, *Differentiation of a porcine reproductive and respiratory syndrome virus vaccine strain from North American field strains by restriction fragment length polymorphism analysis of ORF 5*, Journal of Veterinary Diagnostic Investigation, 10 (1998), pp. 140–144.

[121] R. D. Wesley, W. L. Mengeling, K. M. Lager, A. C. Vorwald, and M. B. Roof, *Evidence for divergence of restriction fragment length polymorphism patterns following in vivo replication of porcine reproductive and respiratory syndrome virus*, American journal of veterinary research, 60 (1999), pp. 463–467.

[122] H. Wold, *A study in the analysis of stationary time series*, 1938.

[123] C. Wu, F. Wu, T. Qi, Y. Huang, and X. Xie, *Noisytune: A little noise can help you finetune pretrained language models better*, (2022).

[124] Z. Wu, S. Wang, J. Gu, R. Hou, Y. Dong, V. G. V. Vydiswaran, and H. Ma, *Idpg: An instance-dependent prompt generation method*, 2022.

[125] L. Xu, M. Skoularidou, A. Cuesta-Infante, and K. Veeramachaneni, *Modeling tabular data using conditional GAN*, Advances in Neural Information Processing Systems, 32 (2019).

[126] M. Yasir, A. M. Karim, S. K. Malik, A. A. Bajaffer, and E. I. Azhar, *Prediction of antimicrobial minimal inhibitory concentrations for Neisseria gonorrhoeae using machine learning models*, Saudi Journal of Biological Sciences, 29 (2022), pp. 3687–3693.

[127] Y. Zhi, Z. Jin, L. Lu, T. Yang, D. Zhou, Z. Pei, D. Wu, D. Fu, D. Zhang, and X. Li, *Improving atmospheric corrosion prediction through key environmental factor identification by random forest-based model*, Corrosion Science, 178 (2021), p. 109084.

[128] Y.-G. Zhu, T. A. Johnson, J.-Q. Su, M. Qiao, G.-X. Guo, R. D. Stedtfeld, S. A. Hashsham, and J. M. Tiedje, *Diverse and abundant antibiotic resistance genes in chinese swine farms*, Proceedings of the National Academy of Sciences, 110 (2013), pp. 3435–3440.

[129] Z. Zhu, T. Liu, and Y. Liu, *A second-order approach to learning with instance-dependent label noise*, in Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), June 2021, pp. 10113–10123.