Master thesis on Intelligent Interactive Systems

Universitat Pompeu Fabra

Taxonomic classification of metagenomic reads using machine learning models

Eduard Alcobé Garcia

Supervisor: Mario Ceresa

Co-Supervisor: Antonio Puertas & Vicenç Gómez

July 2022

Master thesis on Intelligent Interactive Systems

Universitat Pompeu Fabra

Taxonomic classification of metagenomic reads using machine learning models

Eduard Alcobé Garcia

Supervisor: Mario Ceresa

Co-Supervisor: Antonio Puertas & Vicenç Gómez

July 2022

Contents

Dedication

I would like to dedicate this work to my family, especially my parents and my sister.

Acknowledgement

- I would like to express my deepest gratitude to my supervisor Mario Ceresa without whose guidance and support this master thesis would have been not possible. I'm also extremely grateful to Antonio Puertas for his help and interest in how the dissertation was going. Additionally, I would like to extend my sincere thanks to Vicenç Gómez for his advice and recommendations not only in the master thesis but also throughout the master's course.

- Thanks should also go to all the teachers I have had the pleasure to meet during this master. All of them have contributed to my learning and, in their own way, inspired me. I am also grateful to all the members of the Knowledge for Health, and Consumer Safety Unit (F7) of the Ispra centre of the Joint Research Center.

- Lastly, I want to thank my family, especially my parents and my sister, for their unconditional support. They have always given me encouragement and emotional support to pursue my goals.

Abstract

Microorganisms such as bacteria can be hard to identify correctly. Most current classification techniques are based on well conserved genes, for instance the 16S ribosomal RNA (16S rRNA). Nevertheless, achieving a classifier with high accuracy in classifying bacteria through 16S rRNA data is still a challenge. For this reason, different machine learning approaches exploring a k-mer representation technique can still contribute to solve this problem. Mapping the DNA sequences as vectors in a numerical space, by counting the frequency of each k-mer in a given sequence, is essential to be able to train the machine learning algorithms. Two deep learning models, Convolutional Neural Networks and Deep Belief Networks, as well as a tree-based model, XGBoost, are trained with synthetic datasets with 16S rRNA sequences. These synthetic datasets are the 16S rRNA shotgun (SG), and the amplicon (AMP) which considers only specific 16S hypervariable regions. Comparing the performance of these models with the synthetic datasets provides useful information. Moreover, it is also relevant to explore how these models work with real data available in public genomic databases (NCBI, SILVA and FDA-ARGOS). Analysing the classifiers' performance with real data contributes to give an estimation of the reliability of both, classifiers and public genomic databases.

Keywords: Machine learning; Metagenomic; Bacteria; 16S ribosomal RNA; K-mer representation; Classifier; Public genomic databases

Chapter 1

Introduction

Metagenomics is a term that describes a scientific research and a set of techniques that deals with direct genetic analysis of genomes contained with an environmental sample [1]. Metagenomic analysis is an important task for the scientific community as it allows characterising bacterial community composition and to identify microorganisms correctly to know their potential effect on humans [2]. However, it is still a challenge to achieve a complete tool to perform such analysis using well conserved genes [3]. The similarity in those genes may lead to dangerous misclassifications. This fact is much more important when there is confusion between beneficial and harmful microorganisms. [1]

Microorganisms, such as bacteria, are constituted by cells which contains genes made up of DNA. Each specie has a set of well conserved genes, the analysis of which allows to determine its organisms. This is the so-called biological taxonomy [4]. The structure of DNA (Fig.1) can be represented as a double helix composed of base pairs of nucleotides. The four bases comprising the DNA chains are adenine (A), cytosine (C), guanine (G) and thymine (T). The bases that form pairs are A with T, and C with G and vice versa.

Nowadays, the so-called Next-Generation Sequencing (NGS) techniques [5], based on a dataset, facilitate metagenomics reducing the importance of experimental tools by determining directly the whole collection of genes. Moreover, this approach

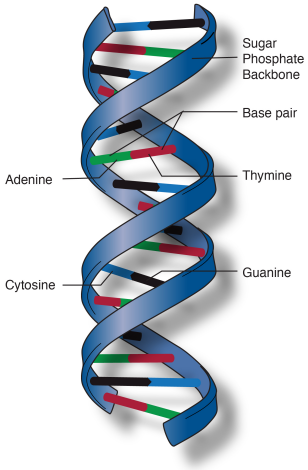

Figure 1: Visualization of DNA double helix structure and pairs of bases[1].

solves the limitation suffered with the traditional method, which has a problem when microorganisms living in community cannot be easily isolated or are difficult to grow in a lab.

The 16S ribosomal RNA (16S rRNA) gene sequence is the most extensively used marker gene for profiling bacterial communities [6]. The 16S rRNA gene is highly conserved, being ideal to identify or classify bacteria. Despite being an RNA gene, the four bases (A, C, G, T) are used when puristically uracil (U) should be used instead of thymine (T). Fig.2 shows how 16S rRNA gene sequence looks like. As it can be seen in Fig.2, the sequences consist of nine hypervariable regions [7]. These regions are highly conserved regions named from V1 to V9. In this work, two NGS technologies regarding the 16S rRNA sequencing are used following Fiannaca et al. [8]. The first is the whole genome shotgun (WGS) which filter the RNA sequence short-read data to obtain short reads (small pieces of sequenced DNA) belonging to a 16S shotgun (SG). The shotgun sequencing method consists in breaking the DNA sequence into small segments, which are sequenced to obtain reads. The second takes into account some hypervariable regions, specifically V3 and V4, and it is called amplicon sequencing technique (AMP). Both datasets used are provided by Fiannaca et al. [8].

[1]Source: `https://www.genome.gov/genetics-glossary/Double-Helix`

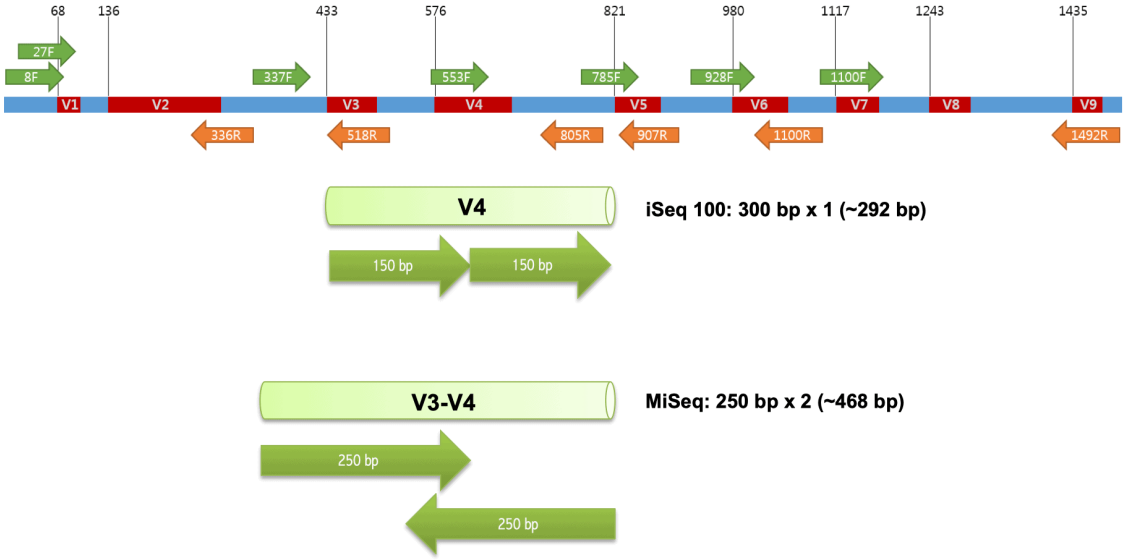

Figure 2: 16S ribosomal RNA (16S rRNA)[2].

After getting the 16S rRNA sequences, a machine learning method may be applied for taxonomic profiling. This is the main point of this work to assess the accuracy of several machine learning methods for bacterial typing. Many pipelines are based on the RDP classifier, which is a Naive Bayesian classifier [9]. However, deep learning techniques like Convolutional Neural Networks (CNN), and Deep Belief Network (DBN) classifiers have been proven better than RDP classifier [8]. These results are reproduced and complemented, in this study. Moreover, another machine learning model is created, specifically a decision tree classifier named XGBoost [10]. Finally, the best models are tested using available public genomic databases.

The work presented in this master thesis has been carried out in collaboration with the *Knowledge for Health, and Consumer Safety Unit (F7)* of the Ispra centre of the *Joint Research Center (JRC)*[3]. The lines of investigation to follow have been jointly decided.

[2]Source: `https://help.ezbiocloud.net/16s-rrna-and-16s-rrna-gene/`
[3]Joint Research Center (JRC)

1.1 Motivation

"You can have data without information, but you cannot have information without data" - Daniel Keys Moran[4].

Big data, machine learning, and artificial intelligence have become the latest buzzwords in many research fields and applications. The amount of possibilities seems nearly limitless due to the variety of problems where machine learning techniques are applied, and the flexibility and adaptability of different methods. For this reason, I find these three subjects fascinating, and incredibly useful. This is the future, although ethical principles have to be respected [11]. Concretely, applying machine learning to process a great number of data in real applications is one of my interests. Some years ago, it was impossible to compare lots of data and to find patterns. Nevertheless, the emergence of machine learning methods makes a major breakthrough on the road to learn from a large amount of data.

The applications of machine learning are focused on areas of engineering, medicine, biology, agriculture, education, business, and social sciences in both, industry and academia [12]. Explicitly in genetics, the use of machine learning has revolutionized the research, allowing humans to extract information from massive amounts of data [13, 14]. Now, the problem is the necessary data mining to extract useful biological information from large datasets [15]. Complex biological problems such as classifying microorganisms according to their DNA are still a challenge due to the nature of the data. Therefore, analysing machine learning or deep learning algorithms as well as efficient strategies to deal with DNA data is a challenge for me. Moreover, working to contribute to a real world problem is an extra motivation to see which is the impact that machine learning can make.

Being able to correctly detect and classify a microorganism has many applications in the medical field [16, 17] but also in other fields such as biotechnology [3, 17]. The amount of possible applications of metagenomics justifies by itself its importance. Not only metagenomic analysis can save lives, and prevent diseases having a quick,

[4]Computer programmer, and science fiction writer

and efficient identification of a microorganism but also may make an impact in many research fields: microbiology [18], ecology [19], bioremediation [20], aquaculture [21], and pharmaceutical industry [22].

The specific motivation behind this master thesis is to classify bacteria efficiently in order to distinguish between harmful and harmless bacteria for humans. Correctly classifying bacteria is essential to alert or not about a potential disease. Historically, more than one example demonstrates the importance of avoiding false positives and false negatives. For instance, false positives for *Mycobacterium tuberculosis* have been analysed [23], or false negatives for *Pneumococcal sepsis* have been studied as well [24]. Another, more recent example is the misidentification of anthrax placed in the New York City subway system [25]. This error was caused due to low-quality reference genomes and shows how essential it is to have a complete and reliable dataset. The lack of reference genomes represents a problem to classify bacteria correctly. For this reason, developing machine learning or deep learning techniques based on a dataset to classify bacteria through DNA data may result in a great method. Furthermore, having such a model would be extremely useful to swiftly analyse a great number of samples at the same time.

1.2 Objectives

The basic aims of this master thesis are to compare various algorithms to classify microorganisms according to the 16S rRNA gene and to provide an estimation on how well the classifiers perform and on how reliable the public genomic databases are. The main objectives are:

1. **Comparison of classifiers' accuracy:** One of the essential goals of this master thesis is to compare the performance of different classifiers in predicting the taxonomy of bacteria through their genome. In this case, CNN, DBN and XGBoost are the analysed models. Calculating the accuracy of each model is key to measure the success of each algorithm, although other metrics like the F1 score are also analysed. In order to compare algorithms, the criterion used

has to be uniformed. Consequently, all the models are trained using the SG and AMP datasets, which do not represent all taxonomic bacteria levels.

2. **Using the best models to evaluate public genomic databases such as NCBI, SILVA and FDA-ARGOS:** The other major goal is to test the best classifiers with real data available in public databases. For this reason, large and complete datasets are required. The datasets used for this purpose are: two datasets formed by data available in the National Center for Biotechnology Information (NCBI) website [26] (see Section 2.1.2); the last high quality dataset released in SILVA database [27] (see Section 2.1.3); and data in FDA-ARGOS database, which is formed by quality-controlled microbial reference genomes [28] (see Section 2.1.4). In other words, the final objective is to be able to predict to which bacterium belongs a DNA sequence, taking into consideration the existent public genomic databases. Fundamentally, the number of mismatches is sought to be as low as possible. The misclassifications can be due to the classifier or because of a wrong label in the corresponding database, therefore understanding them is necessary to assess the reliability of the public genomic databases.

1.3 Background

There exists some literature addressing the problem of classifying microorganisms using machine learning models. One of the basis of this thesis is the paper published by Fiannaca et al. [8] in which CNN, DBN, and RDP techniques are applied for SG and AMP datasets. Moreover, other useful papers are the article where the RDP classifier is presented [9], a guide concerning metagenomics [29], and a review about applied machine learning in metagenomics classification which presents state of the art about computational methods for processing metagenomic data using machine learning, data science, and big data [30].

The use of deep learning for taxonomic classification has also been discussed by Liang et al. [31] where CNN, a Long Short-Term Memory (LSTM), and a hybrid

deep neural network of CNN and LSTM strategies are evaluated. The possibility of using a Recurrent Neural Network (RNN) as an attention mechanism to classify metagenomic sequences has also been explored [32]. Furthermore, similar to taxonomic classification of bacteria, recently the scientific community has focused on the importance of classifying virus [33].

1.4 Structure of the Thesis

This thesis is structured as follows:

1. **Methods:** The followed pipeline is explained. Starting with the datasets, synthetic datasets, SG and AMP, are briefly explained, as well as the datasets downloaded from NCBI, SILVA and FDA-ARGOS databases. Then, the k-mer representation as genomic data representation is mentioned to transform the original data to a numeric sequence suitable for the machine learning or deep learning algorithms. Third, the classifiers used are discussed considering their architecture. Following the description of the machine or deep learning techniques, the metrics calculated to evaluate them are presented. Finally, the experiments carried out to achieve the objectives are mentioned.

2. **Results:** Once the methodology is clear, the results achieved are presented and analysed. The accuracy of each model is the most crucial point, but for instance, the computing time is also studied as well as different other metrics regarding the importance of true positives, true negatives, false positives and false negatives identifications. Moreover, the loss function of the training and validation datasets for different models is analysed. The effect of some parameters for some models is also studied. Then, the classifiers' performances are compared, and the best models are tested with datasets from public genomic databases in order to evaluate their reliability.

3. **Conclusions:** At the end, the main conclusions of the whole research work are presented. Furthermore, the limitations found are mentioned, and future tasks to overcome them are proposed.

Chapter 2

Methods

In this section, the methodology followed is explained. The proposed training pipeline is shown in Fig.3 in order to visually explain the steps for taxonomic classification of metagenomic data.

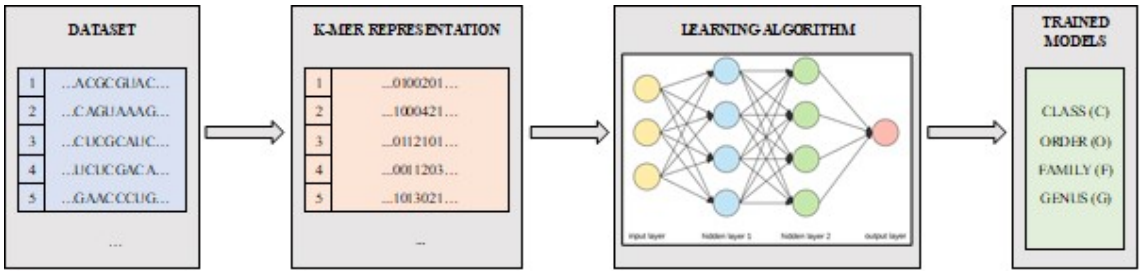

Figure 3: Training process to classify microorganisms. Starting from 16S reads, a vector representation using the frequency of k-mers is performed, and a machine or deep learning architecture is trained to obtain models for taxonomic classification.

Different datasets, SG, AMP, TargetedLoci and Nucleotide16S (from NCBI), SILVA and FDA-ARGOS are used as reads, knowing the taxonomic classification (at least at genus level) of all DNA sequences in these datasets. Nonetheless, it has to be highlighted that the nature and labelling of the datasets is not the same for each of them. With the mentioned data, a vector representation in the form of k-mers frequencies is created to make the dataset suitable for a deep learning algorithm

based on neural networks or a tree-based model. The models used during the learning process are a Convolutional Neural Network (CNN) and a Deep Belief Network (DBN) as deep learning techniques and XGBoost as a tree based method. Fig.4 illustrates the use of each dataset and model. On one hand, SG and AMP in k-mers frequency form are used to train and validate CNN, DBN and XGBoost models. On the other hand, the datasets representing NCBI, SILVA and FDA-ARGOS databases are used to test the best models of CNN, DBN and XGBoost. All details about the mentioned steps are expounded in the following sections.

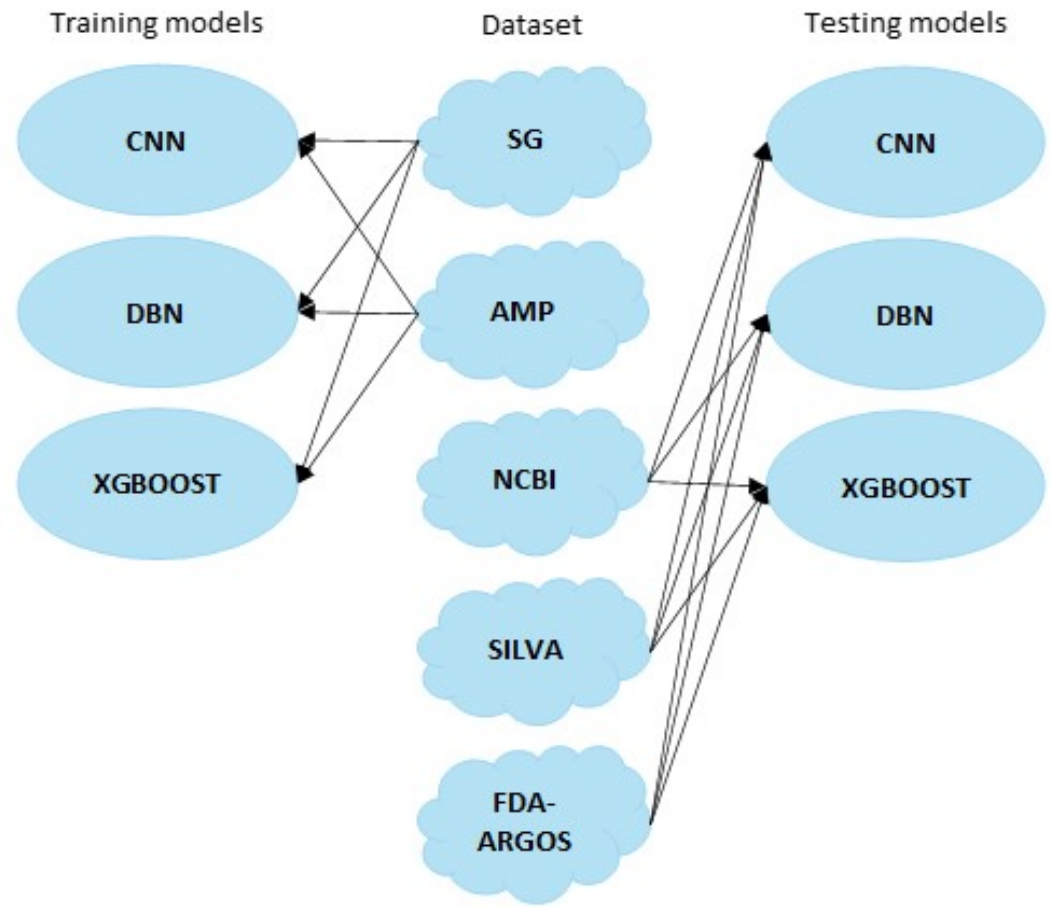

Figure 4: Representation of which dataset is used for each model.

2.1 Datasets

The different datasets used are explained in this section. First, the synthetic datasets called SG and AMP are described. Then, the datasets from NCBI (TargetedLoci and Nucleotide16S) are exposed, followed by the explanation of the SILVA dataset. Finally, FDA-ARGOS dataset, representing a great number of bacteria species, is also presented.

In order to understand the different datasets, it is important to be aware of the bacterial taxonomy. The most common rank from the most generic to the most specific level is: domain, phylum, class, order, family, genus, and species. For example, the bacterial taxonomy for the *Legionella Pneumophila* is: domain *Bacteria*; phylum *Proteobacteria*; class *Gammaproteobacteria*; order *Legionellales*; family *Legionellaceae*; genus *Legionella*; species *Legionella pneumophila*.

2.1.1 SG and AMP

The synthetic or artificial datasets named SG and AMP are short-reads sequences provided by Fiannaca et al. [8] that have been used in their publication[1]. Short-reads in the datasets are small pieces of sequenced DNA. Such datasets were built following the pipeline or approach used by Yuan et al. [34] and Ramazzotti et al. [35]. These datasets are composed of thousand of 16S rRNA sequences. Specifically, the SG dataset has 28224 sequences and the AMP 28000. All these sequences are labelled, which is a requisite to train a supervised classifier model later. Regarding the taxonomic levels, for the SG dataset, 3 different classes are represented, 20 orders, 39 families and 100 genus. Each of the genus is represented at least with 250 different 16S rRNA sequences. However, for the AMP not all genus are present, but 96 of them. For this reason, the taxonomic classification at class level is the easier task, while at genus level may be hard.

[1]Source: `http://tblab.pa.icar.cnr.it/public/BMC-CIBB_suppl/datasets/`

2.1.2 NCBI: TargetedLoci and Nucleotide16S

The National Center for Biotechnology Information (NCBI) has many genetic data in their databases [26]. Thereby, it is possible to extract a collection of bacteria genomes and to create a very complete dataset to test the models. As mentioned, NCBI has a lot of information, for instance, there is the Reference Sequence (RefSeq) collection which provides a comprehensive, integrated, non-redundant, well-annotated set of sequences, including a dataset of 21626 different 16S rRNA sequences of bacteria[2]. This dataset is named TargetedLoci. Obviously, when testing the models using this dataset, only the sequences of bacteria known by the model are used. Therefore, the dataset has to be filtered to keep the sequences of the 100 genus of bacteria present in the synthetic SG dataset. For this reason, the final number of useful sequences for the present report is 1330. Another dataset have been downloaded through NCBI website using their nucleotide database[3] and filtering by bacteria as species, rRNA as molecule types and 16S in the title. In total, 39361 16S rRNA sequences of bacteria are in this dataset of which 1475 are susceptible to be used in the testing step. This second dataset will be called Nucleotide16S. In both datasets from NCBI, each genomic sequence is labelled with the corresponding bacterial genus and species.

2.1.3 SILVA

SILVA is a high quality ribosomal RNA database belonging to the German Network for Bioinformatics Infrastructure (deNBI) [27]. SILVA database offers interesting tools apart from subunit ribosomal RNAs such the 16S. For instance, they have a TestProbe tool to find sequences along their databases. Specifically, the dataset used to test the trained models is named SILVA SSURef NR99 from the 138.1 release, but in this report it will be referred to simply as SILVA[4]. This dataset is composed of 16S rRNA sequences of archaea and bacteria, thereby it will be filtered to keep just the bacteria sequences and what is more, just the bacteria known by the trained

[2]Source: `https://ftp.ncbi.nlm.nih.gov/refseq/TargetedLoci/Bacteria/`
[3]Source: `https://www.ncbi.nlm.nih.gov/nuccore`
[4]Source: `https://www.arb-silva.de/no_cache/download/archive/release_138.1/`

models. It may be highlighted that this dataset has the full taxonomy for each of the bacteria species. Nonetheless, the interesting taxonomic rank is the bacterial genus, thereby the filtering process takes that into account. The total amount of sequences is 510508, however after filtering the number is reduced to 13729.

2.1.4 FDA-ARGOS

FDA-ARGOS is a very complete database with public quality-controlled reference genomes for diagnostic purposes [28]. It is a hand-cured dataset which assures high quality DNA sequences. The aim of this dataset was to fill the gap in the public domain of reference genomes. Specifically, it is focused on known relevant microorganisms that can lead to a disease. The amount of different genomes present in the first subset of FDA-ARGOS dataset was 487, however not all were bacteria, but the 88.3%, thereby the number of bacteria genomes was 430 representing 76 different bacterial genus. However, FDA-ARGOS is continuously updating, therefore the number of different genomes is now 1428 with a total number of sequences larger than 5 millions since for each of the 1428 FDAARGOS genomes there is the whole genomic sequences, specific protein sequences and RNA information[5]. In this work, only the 16S ribosomal RNA sequences are of interest and in FDA-ARGOS there are 9939 16S rRNA sequences representing 145 different bacteria. However, after filtering the data to keep only the bacteria known by models trained with a synthetic dataset, the number of sequences decrease to 225 of 8 different bacterial genus (*Aggregatibacter*, *Brevundimonas*, *Kingella*, *Legionella*, *Neisseria*, *Rhizobium*, *Sphingomonas* and *Stenotrophomonas*).

2.2 Genomic data representation

DNA sequences are composed by nucleotides (A, C, G, T), however this is not the ideal kind of data to train a deep learning model. It is necessary to have useful features from the data to extract information and patterns when learning. For this reason, the k-mers strategy is often used to map DNA sequences as vectors in a

[5]Source: `https://www.ncbi.nlm.nih.gov/bioproject/231221`

numerical space. This means that DNA sequences are converted in to a numerical space using the number of times that a k-mer appears in each of the given sequences. In other words, the frequency of all the possible combinations of the four nucleotides of k length is used, losing the information of where that nucleotides were in the sequence. Each k-mer acts as a feature, so there are as features as possible combinations 4^k. For instance, following a 3-mers strategy and given $ACATGACA$ as a sequence; the result would be 2 for ACA triplet, 1 for CAT, 1 for ATG, 1 for TGA, 1 for GAC and 0 for all the other possible combinations. One of the main challenges of k-mers is to determine the best value of the k-parameter. A small value may not be representing the pattern good enough, while a high value may be getting only the general structure, losing details. Moreover, the computational time has to be taken into account as the dimension of the defined coordinate space is 4^k. Finding the trade-off between covering the maximum information content and having a practicable computational complexity is not simple.

Once the file representing data using k-mers is created, it is necessary to label every sequence of numbers to have suitable data to be trained. For the SG and AMP datasets, there will be a file for the following taxonomic ranks: class, order, family and genus. However, for the testing datasets, TargetedLoci, Nucleotide16S, SILVA and FDA-ARGOS, only genus is of interest since the aim is to distinguish bacteria at genus level.

2.3 Machine learning methods as classifiers

Two of the classifiers studied are deep learning based, those are a Convolutional Neural Network (CNN) and a Deep Belief Network (DBN), however a tree-based machine learning algorithm such as XGBoost is also considered. Both deep learning architectures were already available [8] while the XGBoost model has been created from scratch. Thereby, the hyperparameters used in CNN and CBN are the same as in the publication of Fiannaca et al. [8] whereas those in XGBoost has been chosen.

Basically, the SG and AMP datasets are split in training and validation sets (no

testing set). This is because external datasets are used to test the models with real rather than synthetic sequences. A 10% of the whole datasets are used to validate the model. Moreover, a 10-fold cross-validation technique is used to achieve a better estimation of the classifiers' performance. Meanwhile, all the possibles filtered sequences extracted from NCBI, SILVA and FDA-ARGOS databases are used to test the best models of CNN, DBN and XGBoost.

2.3.1 Convolutional Neural Network (CNN)

Convolutional Neural Network (CNN) is a deep learning technique often used to analyse and classify images [36]. Nonetheless, CNN can be used to classify any kind of data if converted into the required dimensions. It is based on a layer-per-layer architecture with shared-weights using kernels as filters to extract the features' information. The use of kernels is computationally efficient as it is based on dot products. As mentioned, the architecture of a CNN model is a combination of layers, as a general rule, the number of tuned hyperparameters depends on the number of layers. In this specific case, the only studied hyperparameter is the kernel size, while other arguments are unchanged. Taking into account, other work [8], an architecture combining 1-dimensional Convolutional Layer with Rectifier Linear Unit (ReLU) as activation function and maxpooling layer to reduce the dimensions have been proven efficient. The illustration of the baseline structure of the CNN model created is presented in Fig.5 and explained below:

- 1-dimensional Convolutional Layer with ReLU activation.
- Maxpooling Layer with pool size 2
- 1-dimensional Convolutional Layer with ReLU activation.
- Maxpooling Layer with pool size 2
- Flatten
- Dropout

- Fully Connected Layer with softmax activation.

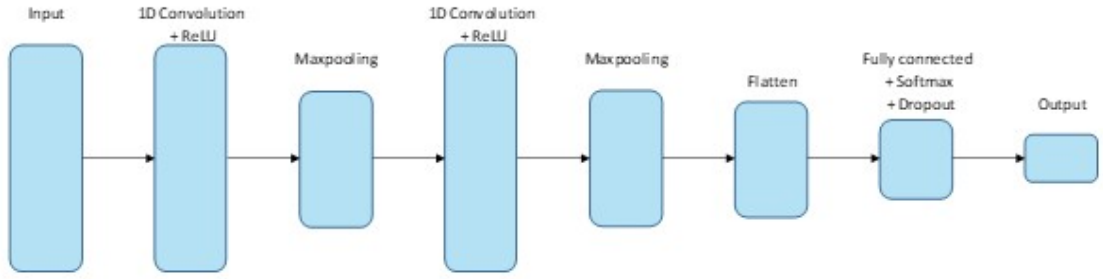

Figure 5: Simplified illustration of the Convolutional Neural Network architecture.

The first convolutional layer has as hyperparameters 5 filters and a kernel size of 5. Meanwhile, the second convolutional layer has 10 filters and a kernel size equal to 5. After combining 1-dimensional Convolutional Layer and Maxpooling Layer, the flatten layer is introduced to reduce the dimension. This step is necessary to facilitate the task of the Fully Connected Layer. Then, a dropout of 0.5 is applied to reduce overfitting, and finally the Fully Connected Layer, with a softmax activation function and the Adam algorithm as optimizer, does the prediction to return a group label as output. Finally, it has to be mentioned that 100 epochs and a batch size of 500 are used during the fitting process.

2.3.2 Deep Belief Network (DBN)

Deep Belief Network (DBN) is a deep neural network based on a graph model approach [37], or in other words, it is a probabilistic model to extract hierarchical representation of the given data. The goal of this algorithm is to learn in probabilistic terms in order to be a useful classifier. Then, a DBN is based on a connected stack of Restricted Boltzmann Machines (RBM) and a Feed Forward Network (FNN) to predict. An RBM is theoretically an unsupervised neural network divided in the so-called visible and hidden layer. The hidden layers neurons are not connected to each other, being conditionally independent, so there are no intralayer connections. Furthermore, it may be highlighted that the training is performed layer by layer. The specific aim of the RBM, in a DBN network, is to obtain a representation of

an input in a lower dimensional space. Using two consecutive RBM, it is possible to estimate a probability distribution of the original input. Fig.6 illustrates the structure of the DBN model.

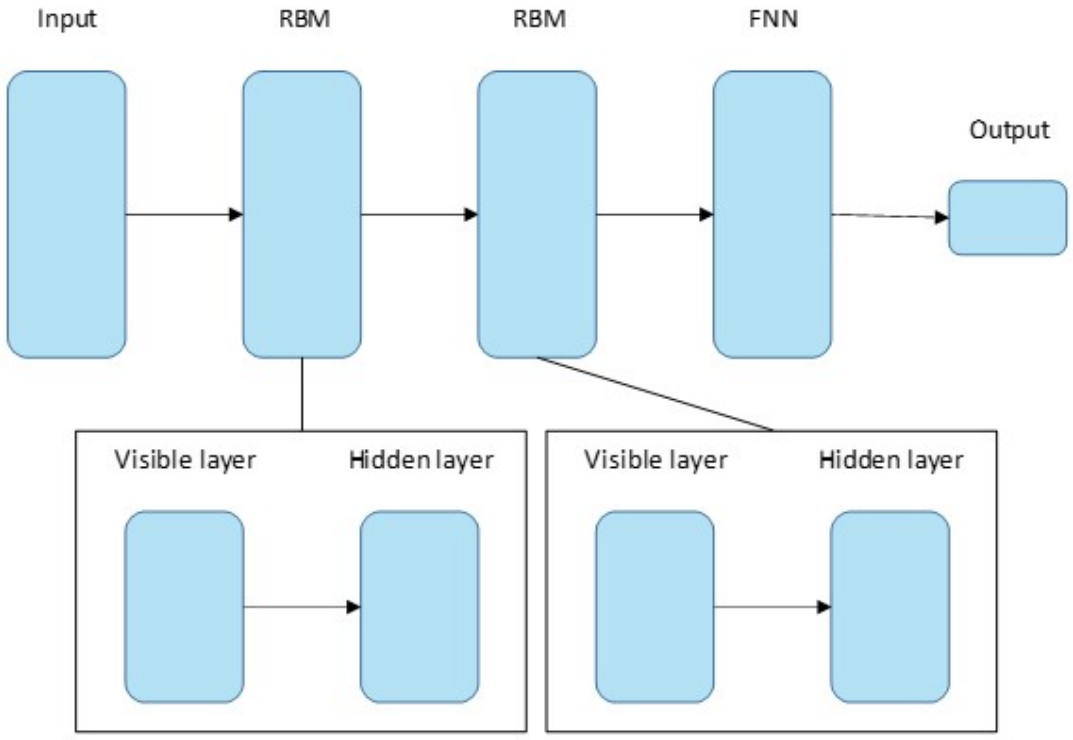

Figure 6: Simplified illustration of the Deep Belief Network architecture.

In the used DBN there are two RBM, whose hidden layers have dimension 256, the learning rate of the RBMs are fixed to 0.05 while for the FNN is 0.1. The activation function to perform the prediction is ReLU with a dropout of 0.2. Therefore, the prediction task is performed by no more than a logistic regression layer acting as a supervised classifier using gradient descent as backpropagation. This layer is the FNN mentioned above. Finally, it has to be mentioned that 100 epochs and a batch size of 32 are used during the fitting process.

Lastly, it is important to highlight that the code used to train the DBN does not support GPU in all the steps. Indeed, the fine-tuning part, which is the one that needs more time, runs in CPU. Hence, the computing time is expected to be high.

2.3.3 XGBoost

XGBoost is a tree-based algorithm based on Gradient Boosting. Decision trees create a model to find the most suitable label by evaluating true/false questions and estimating the probability of having a label or another. XGBoost technique consist in minimizing the loss function by adding other weak models in form of tree and using a gradient descent optimizer. The idea of improving a single weak model by combining it with another weak model is, exactly, the idea of boosting. In this case, the tune hyperparameter is the maximum tree depth. Fig.7 illustrates the structure of the XGBoost model.

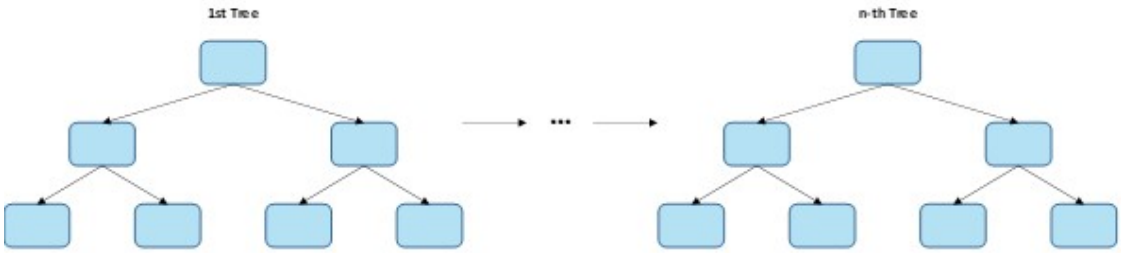

Figure 7: Simplified illustration of the XGBoost architecture.

As it can be seen in the image above, the XGBoost model used is a decision tree with a maximum tree depth of 3. This hyperparameter has been decided taking into account that the method use boosting and therefore shallow trees are sufficient. The weights and the tree are updated from wrongly classified observations. Consequently, the weights and the trees change during the learning process until they reach convergence. 100 epochs are used during the training step.

2.4 Metrics to rank models

Being able to measure the performance of each classifier is essential to discuss which one may be better, hence defining some metrics is a requirement. The first metric considered is the accuracy. It is defined as the number of classifications a model validates correctly divided by the total number of validations made, $\frac{tp+tn}{tp+fp+tn+fn}$ where tp means true positives, fp false positives, tn true negatives and fn false

negatives. Then precision, recall, F1-score and Area Under the Curve (AUC) are also calculated. The precision is the number of true positives during the validation process divided by the sum of true positives and false positives, mathematically $\frac{tp}{tp+fp}$. Whereas the recall quantifies the ratio between true positives and the sum of true positives and false negatives, $\frac{tp}{tp+fn}$. The F1-score can be interpreted as the harmonic weighted mean of both precision and recall, the formula is $2*\frac{Precision*Recall}{Precision+Recall}$. Precision, recall and F1 score are computed weighting each label with their number of actual instances, thereby accuracy and weighted recall are the same. Then, the AUC computes the area under the receiver operating characteristic curve from prediction scores, in other words, the area below the curve plotting the true positive rate, $\frac{tp}{tp+fn}$, vs the false positive rate, $\frac{tn}{tn+fp}$. This is a performance measurement that reports the extent to which the model is able to distinguish between classes, thereby, the highest AUC the better. Finally, some confusion matrix can be drawn in order to represent the most interesting groups of the best models and visually interpret the number of true positives, true negatives, false positives and false negatives.

2.5 Experiments

1. **SG and AMP performance for each classifier:** SG and AMP datasets performance can be compared in order to determine if one NGS technique is better than the other for some of the models to achieve the best possible taxonomic classification. Moreover, some classifiers may be more sensitive to the dataset used (SG or AMP) whereas others may not present a significant difference. Both datasets are taken into account to determine how well an algorithm performs.

2. **Importance of k-mer size:** The strategy followed in this thesis is k-mer based. Thereby, taking into consideration the size of the k-mer representation is necessary to facilitate the classifier task. Having a high k-mer size implies having more features than with a lower k-mer size, consequently the sequences may be easier to distinguish, as there are more features to check. Comparing different k-mer size may help to discern which classifier is better, as some may

be more sensitive to the k-mer size. This experiment is done for the CNN, DBN and XGBoost classifiers using the SG and AMP datasets.

3. **Importance of hyperparameters:** Some parameters of each model are changed to analyse their impact. For instance, in the case of the CNN model, the size of the kernels may affect the accuracy. Depending on this parameter, more detailed information is extracted, or more general knowledge is obtained. For the XGBoost model, the effect of tree depth hyperparameter is explored. In the case of the DBN, no parameter is analysed, hence just the standard created model is used.

4. **Testing the best models of CNN, DBN and XGBoost with public genomic databases:** The best classifiers, according to the experiments already described, have to be tested with more complete and real datasets to be able to predict, with high success, any bacteria through their genome. Logically, all taxonomic levels to be predicted have to be represented in the training dataset. For this reason, the selected datasets have to be filtered in order to use just sequences of bacteria that the models know. Those testing datasets are TargetedLoci, Nucleotide16S, SILVA and FDA-ARGOS. This experiment is ideal to be able to give an estimation of the reliability of public genomic databases.

Chapter 3

Results

The principal objective of this work is to compare the accuracy of different models. Nonetheless, accuracy may vary significantly depending on the k-mer size, the training dataset (SG or AMP) and the taxonomy rank as well as some other specific parameters of each algorithm. The baseline of this work has been the Convolutional Neural Network (CNN) and the Deep Belief Network (DBN) [8], however a XGBoost model is also created and used to compare deep learning methods and non-deep learning algorithms. All three algorithms are trained for different k-mers applying cross-validation with 10 folds.

Apart from the accuracy; precision, recall, F1 Score and Area Under the Curve metrics are calculated. The exact values of these metrics for each of the models can be seen in Appendix A. Furthermore, some confusion matrix for specific models are also shown to understand which groups are harder to classify, especially at the taxonomic rank of genus. Moreover, the learning curves showing the training and validation loss curves for the most promising classifiers are drawn to analyse possible underfitting or overfitting. The loss function applied is the cross entropy, also known as log loss, $H(X) = -\sum_x p(x) \log p(x)$. In addition, specifically for the CNN and XGBoost model, some hyperparameters (kernel size and maximum tree depth respectively) are briefly analysed. All the metrics together are useful to compare the models and discern which model performs the best for each dataset.

Finally, the best models at genus level are tested with external public genomic databases (NCBI, SILVA and FDA-ARGOS) in order to evaluate the quality of both the created classifiers and the public genomic databases.

3.1 Convolutional Neural Network (CNN)

The basic CNN used has a sequence of convolutional and max pooling layers. On one hand, the first convolutional layer has as hyperparameters 5 filters and a kernel size of 5. Meanwhile, the second convolutional layer has 10 filters and a kernel size equal to 5. On the other hand, max pooling layers have size 2.

3.1.1 Accuracy

The accuracy depending on the k-mer size using the SG and AMP datasets for the four different taxonomic ranks is shown in Fig.8 and Fig.9 respectively. In Table.A.1 and Table.A.2 the exact values for the SG and AMP dataset respectively are presented. Significantly, it can be observed how the SG dataset is much more sensible to the k-mer size than the AMP. For the SG dataset, the model does not even reach convergence regarding the accuracy, while for the AMP dataset there is not much gain when using 7-mers with respect to 6-mers. Furthermore, the accuracy clearly increases with the k-mer size. Logically, the class is the easiest taxonomic level to predict as there are just 3 groups in the learning datasets, while genus is the hardest as there are 100 groups.

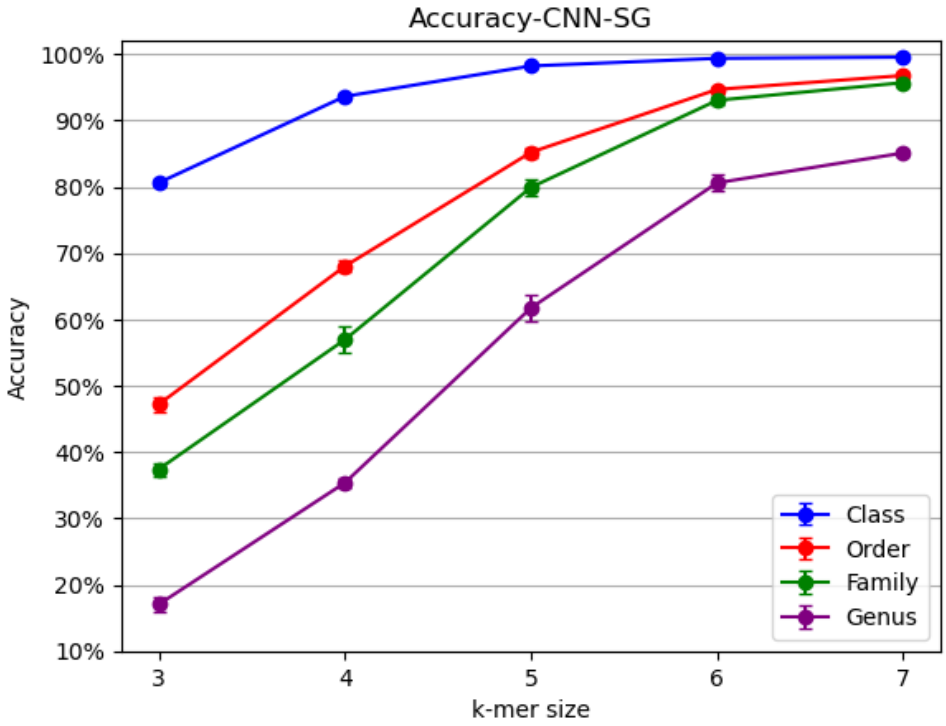

Figure 8: Accuracy of the CNN model with SG dataset.

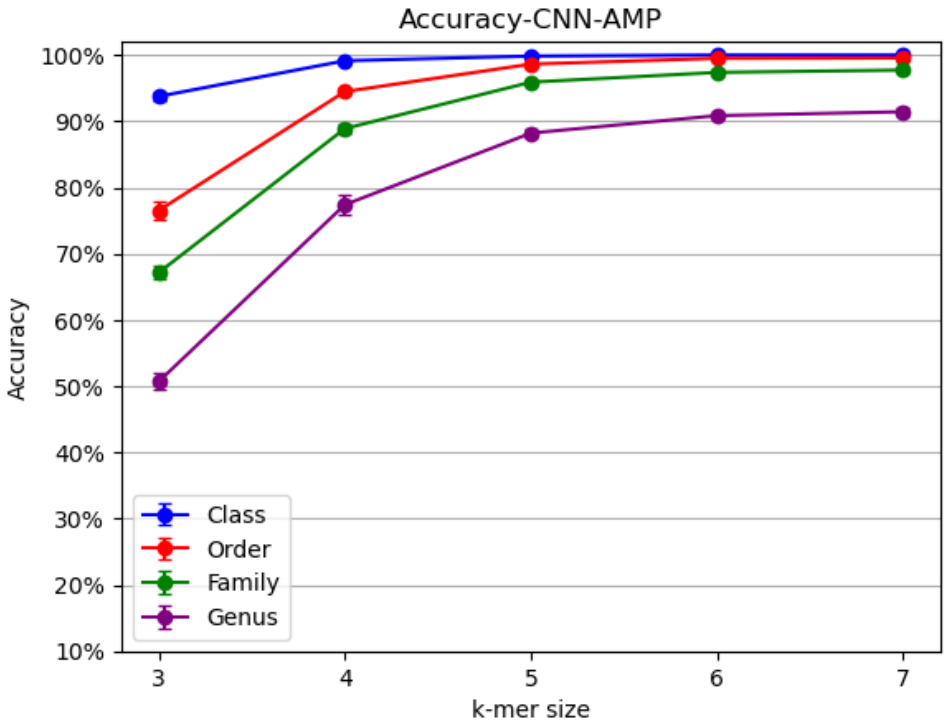

Figure 9: Accuracy of the CNN model with AMP dataset.

3.1.2 Computing time

Regarding the computing time, training the model takes a bit more than an hour (all folds included) for the SG dataset and 7-mers (Table.A.3 shows the exact values). Mostly the same time is needed in the case of the AMP dataset (Table.A.4), although all models trained with AMP takes a little less time. Interestingly, training the model for class, order, family or genus do not have a notable difference in terms of training time. In contrast, the k-mer size has a moderate impact, since the number of features is equal to the number of k-mer, and increases exponentially as 4^k. This behaviour can be observed in Fig.10 and Fig.11. Moreover, comparing the inference time using the SG dataset represented in Table.A.5, and AMP in Table.A.6, it can be seen that using the AMP dataset the inference step is faster than with the SG. Therefore, both, training and inference times are in tune with each other, showing that the CNN algorithm training step is faster with the AMP dataset than with the SG.

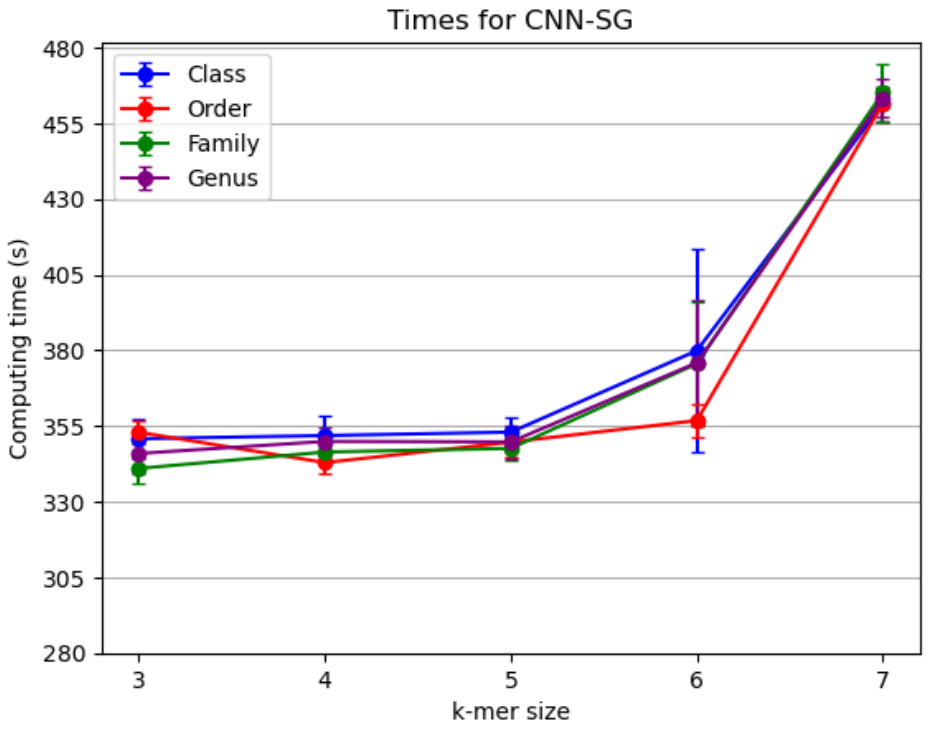

Figure 10: Computing time (s) for the CNN model and SG dataset vs the k-mers size.

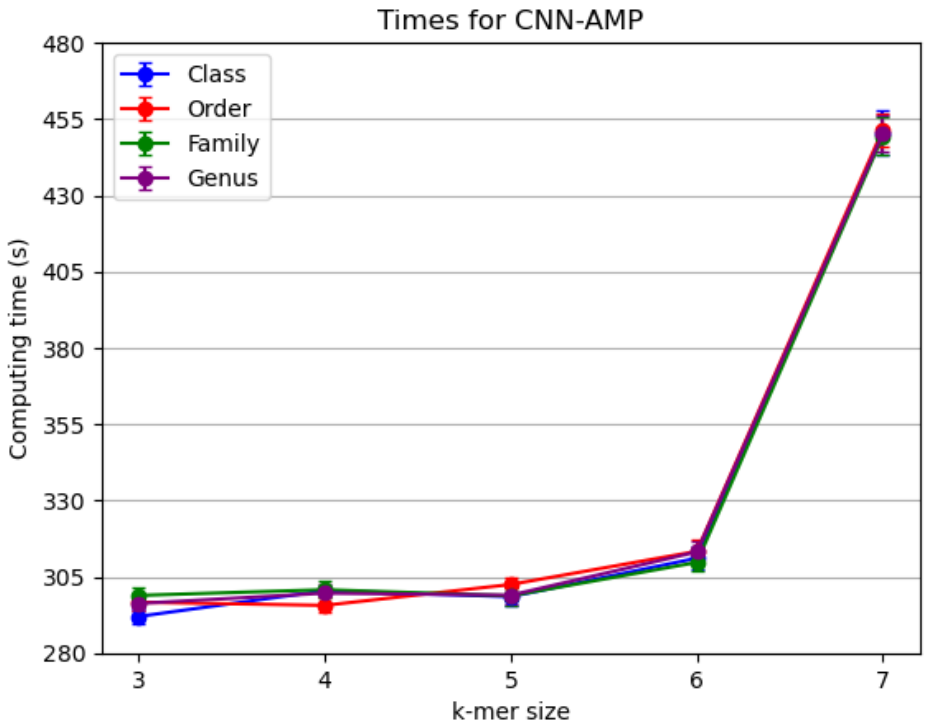

Figure 11: Computing time (s) for the CNN model and AMP dataset vs the k-mers size.

3.1.3 Other metrics

The other calculated metrics have been precision, recall, F1 score and Area Under the Curve (AUC). Fig.12 and Fig.13 shows the evolution of the precision vs the k-mer size when the CNN model has used the SG and AMP datasets respectively. The equivalent results for the recall are presented in Fig.14 and Fig.15. The F1 score is shown in Fig.16 and Fig.17, while for the AUC metric the results are in Fig.18 and Fig.19. Systematically, it can be observed that the greater the k-mer size, the best the model performs. However, interestingly, the AMP dataset is slightly better than the SG considering all the metrics. The same was observed with the accuracy. Furthermore, no significant difference is appreciated between 6-mers and 7-mers when the AMP dataset is used. The specific values for precision, recall, F1 score and AUC can be seen in Appendix A in Table.A.7, Table.A.8, Table.A.9 and Table.A.10.

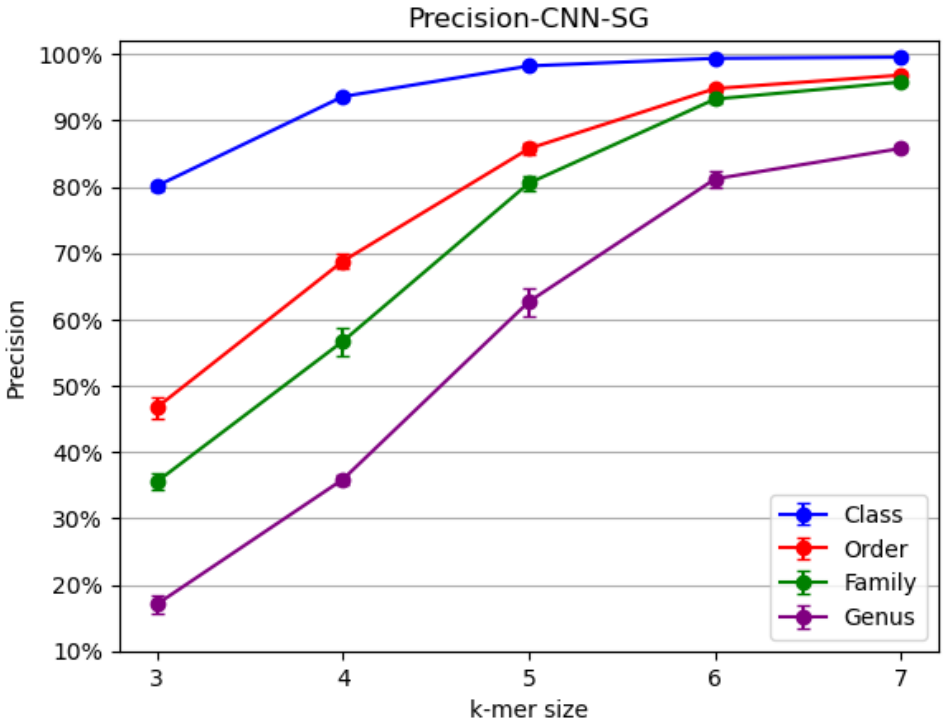

Figure 12: Precision of the CNN model with SG dataset.

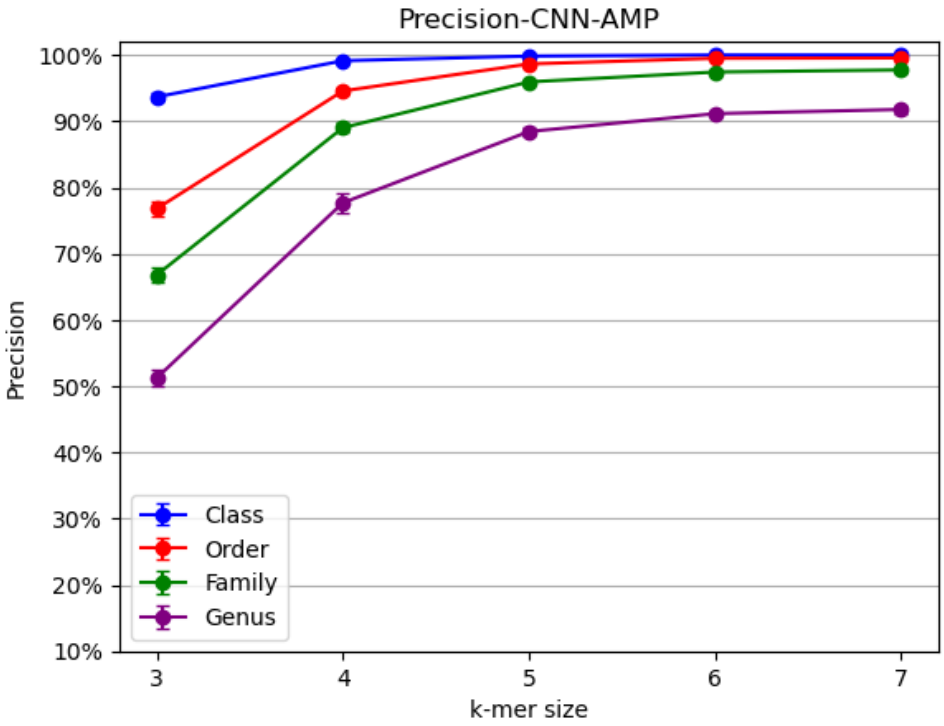

Figure 13: Precision of the CNN model with AMP dataset.

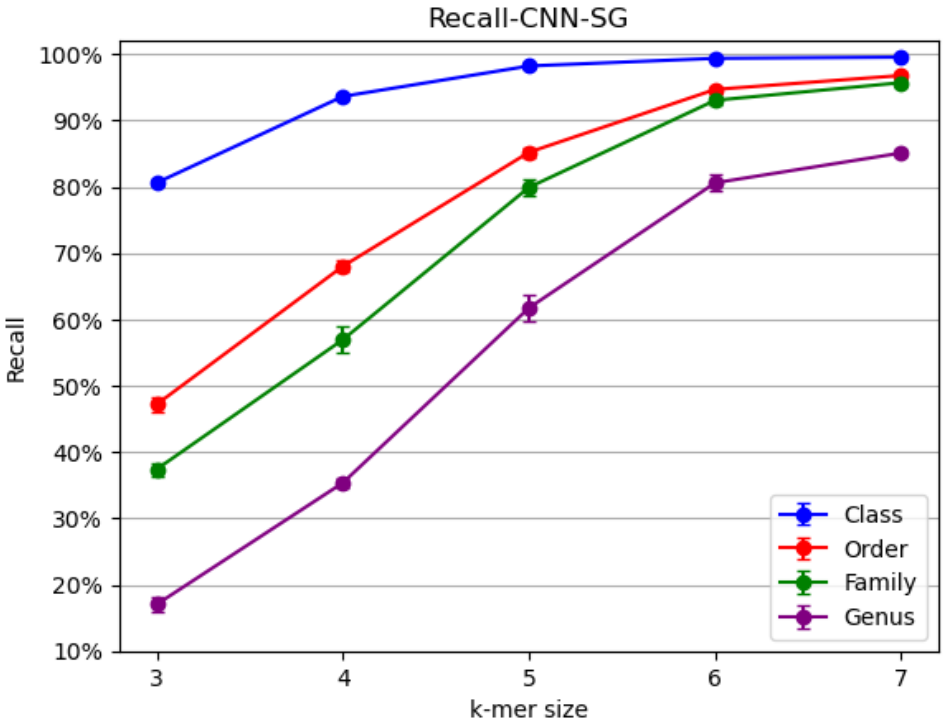

Figure 14: Recall of the CNN model with SG dataset.

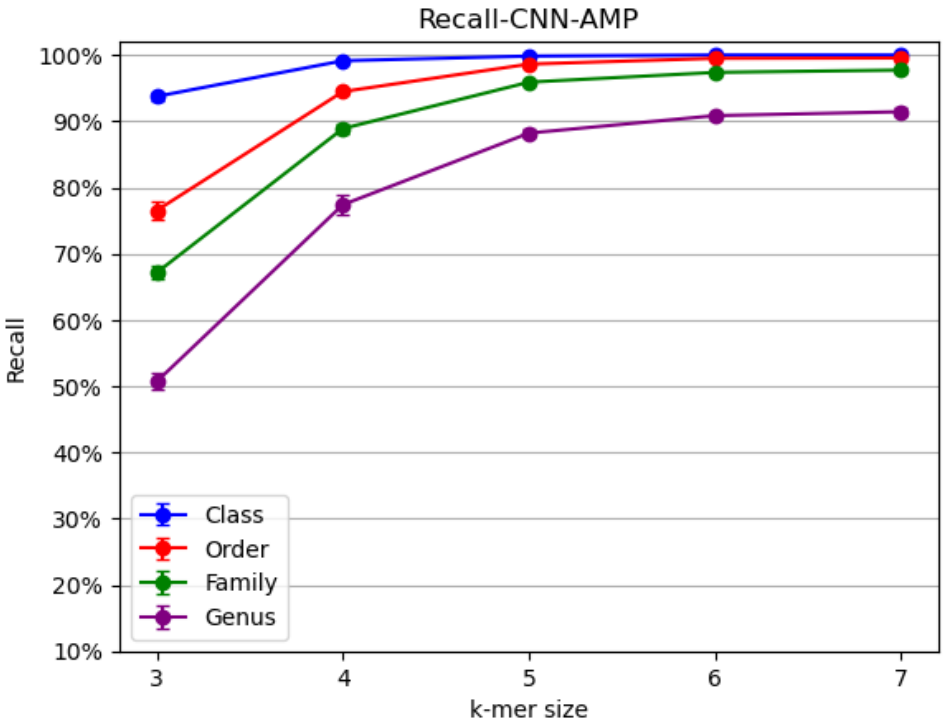

Figure 15: Recall of the CNN model with AMP dataset.

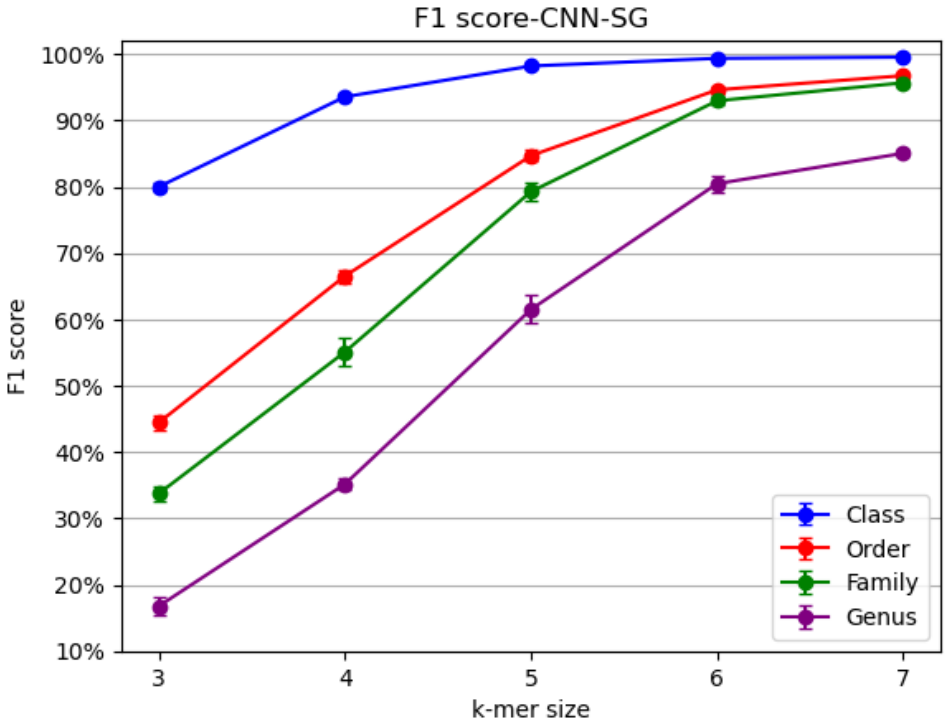

Figure 16: F1 score of the CNN model with SG dataset.

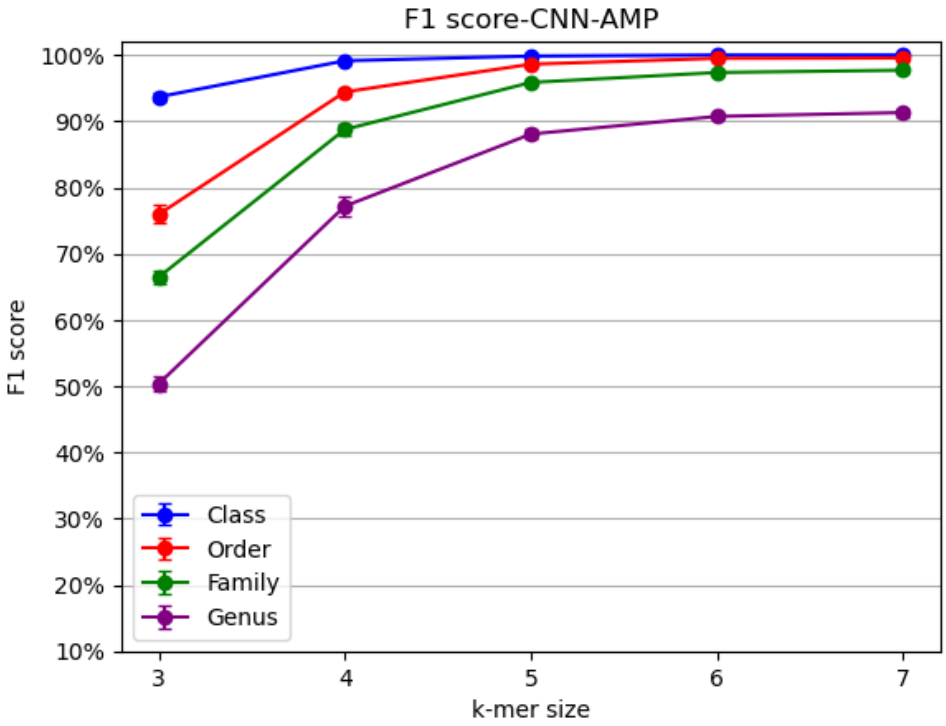

Figure 17: F1 score of the CNN model with AMP dataset.

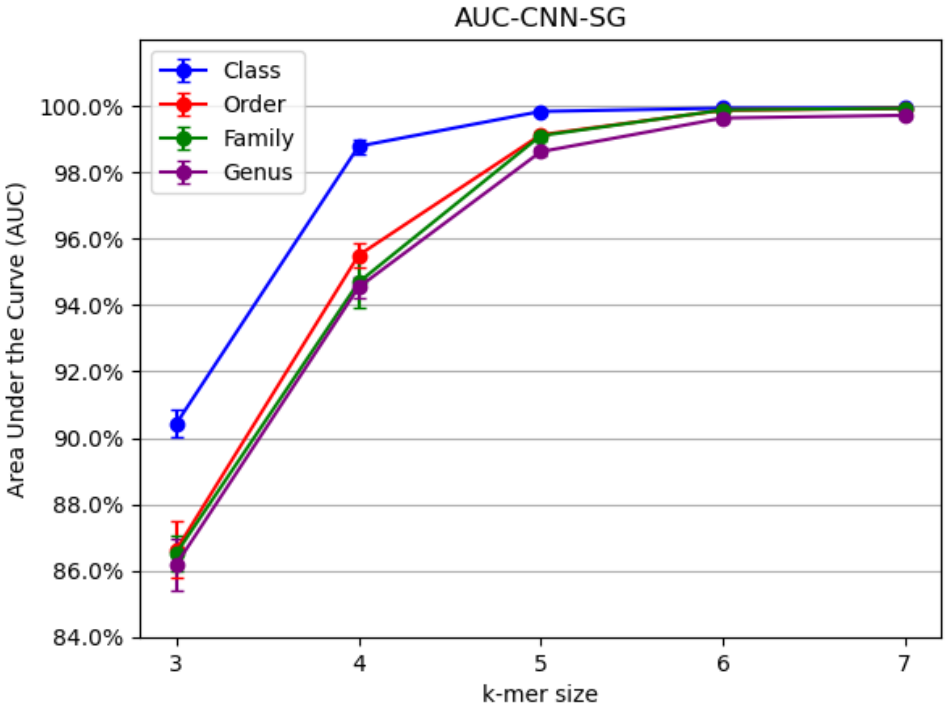

Figure 18: Area Under the Curve (AUC) of the CNN model with SG dataset.

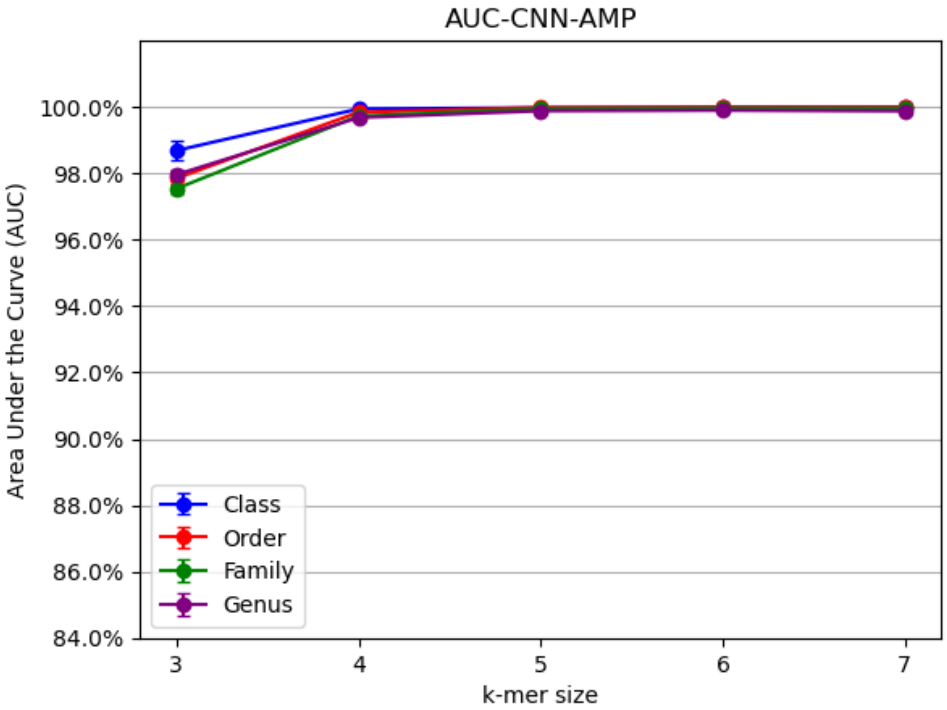

Figure 19: Area Under the Curve (AUC) of the CNN model with AMP dataset.

3.1.4 Confusion matrix

Considering the model trained with the SG dataset and 7-mers as representative, some confusion matrices may be drawn taking into account one of the 10 folds done. The most important taxonomic rank is the genus, since the testing will be performed with the models trained at genus level. For this reason, only the confusion matrix at genus level is shown in Fig.20. The 10 groups with more misclassification appear in it. Meanwhile, the other confusion matrix at class (Fig.A.1), order (Fig.A.2) and family level (Fig.A.3) can be seen in the Appendix A.

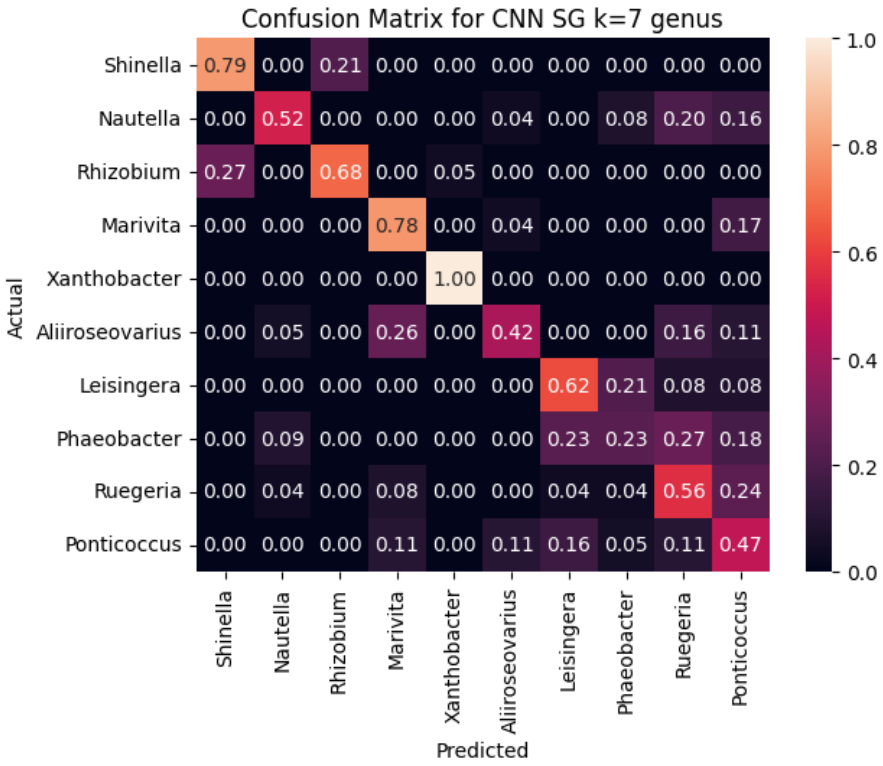

Figure 20: Confusion matrix of the CNN model at the genus level, trained with the SG dataset and 7-mers. Just the genus with more misclassifications are shown.

3.1.5 Loss curves

Analysing the training process of the CNN model through learning curves is essential to understand how the model have learned. For this reason, one fold of two representative models are selected. The first one trained with the SG dataset and

following the 7-mers strategy, and the second one trained with the AMP and 6-mers. For both of them, the training and validation loss curves at class, order, family and genus level are represented to analyse underfitting or overfitting problems. For the model trained with the SG dataset, it is possible to observe the loss curves in Fig.21, Fig.22, Fig.23 and Fig.24. The model trained for the taxonomic rank of genus seems to suffer little overfitting, while for the other easier taxonomic levels the algorithm do not suffer that problem. For the other model, trained with the AMP dataset, the curves may be seen in Fig.25, Fig.26, Fig.27 and Fig.28. The overfitting problem is more severe this time at genus level. In general, it may be highlighted that the CNN classifier learns all the important parameters within few epochs. It has also been observed that the batch size used is essential to avoid noise, since using a small batch size led to really noisy observations, specially in the validation loss. However, using a batch size of 500 such in all graphs presented, there is not much noise.

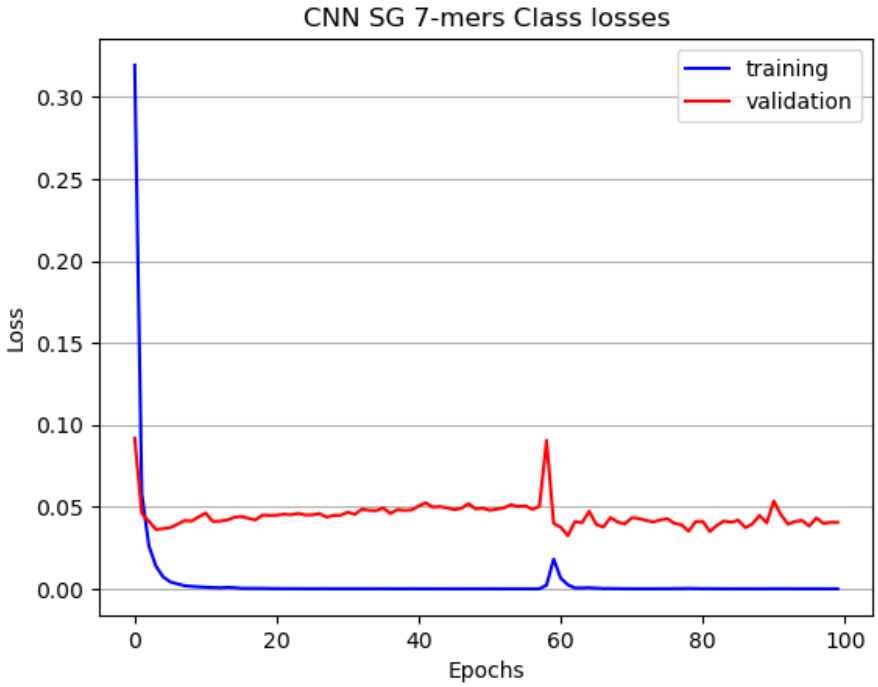

Figure 21: Training and validation loss curves for the CNN classifier tested with the SG dataset and 7-mers at class taxonomic rank.

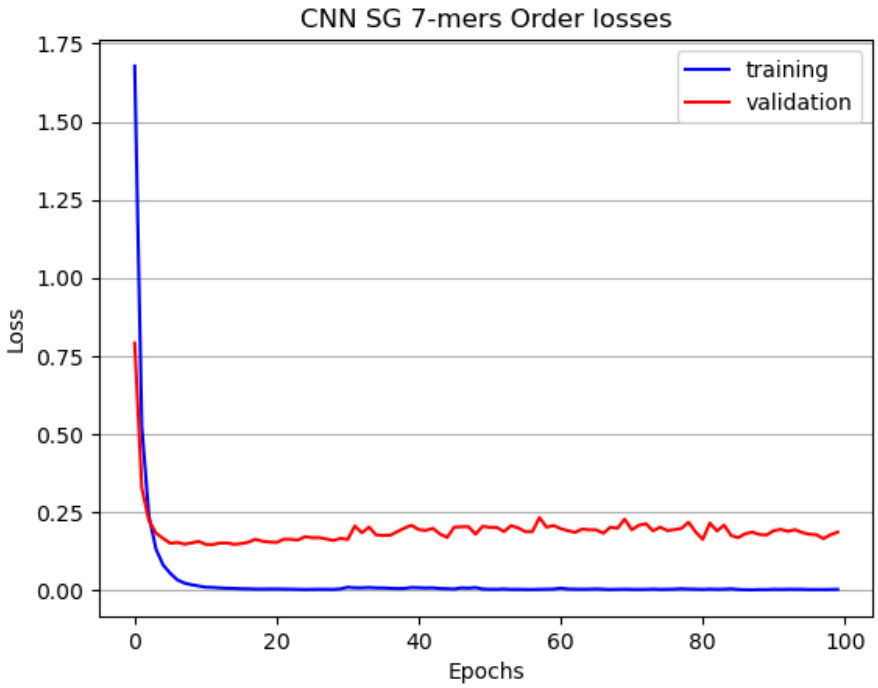

Figure 22: Training and validation loss curves for the CNN classifier tested with the SG dataset and 7-mers at order taxonomic rank.

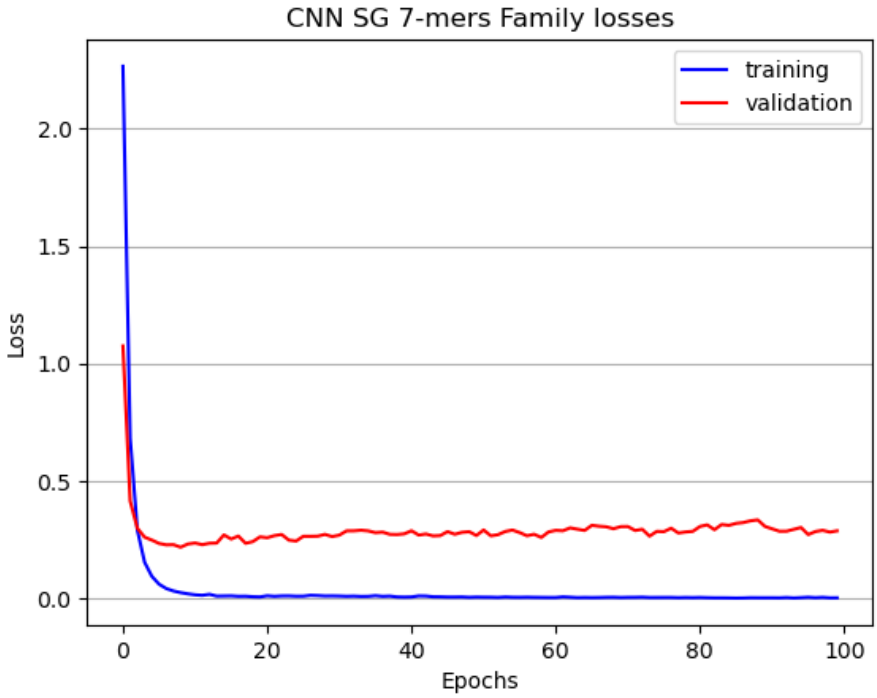

Figure 23: Training and validation loss curves for the CNN classifier tested with the SG dataset and 7-mers at family taxonomic rank.

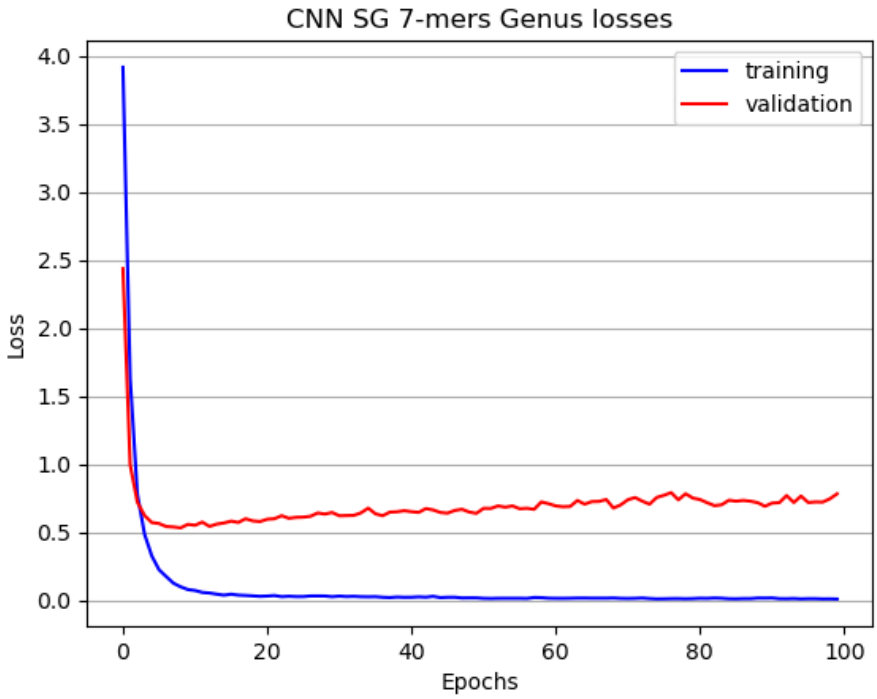

Figure 24: Training and validation loss curves for the CNN classifier tested with the SG dataset and 7-mers at genus taxonomic rank.

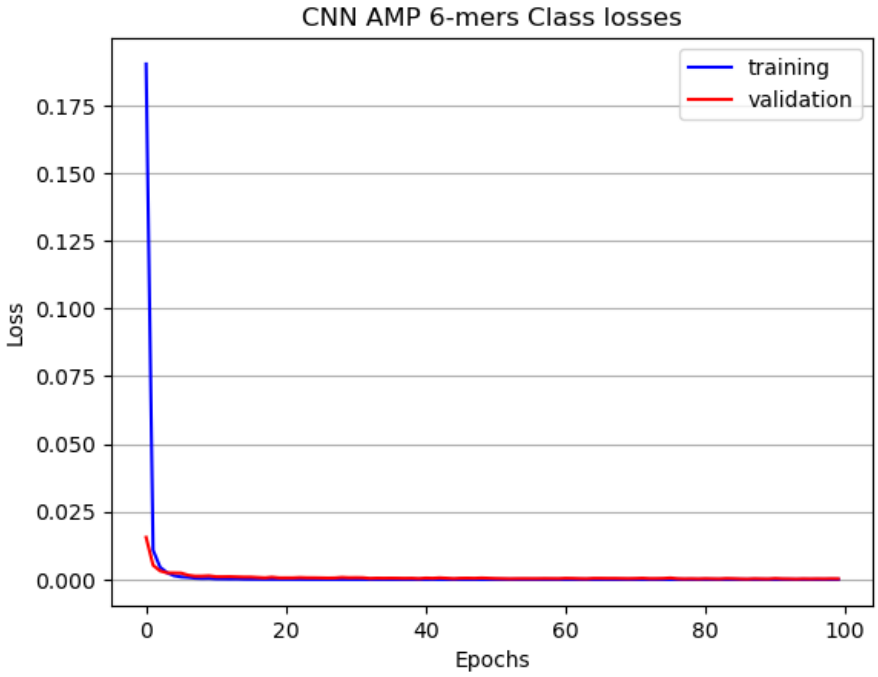

Figure 25: Training and validation loss curves for the CNN classifier tested with the AMP dataset and 6-mers at class taxonomic rank.

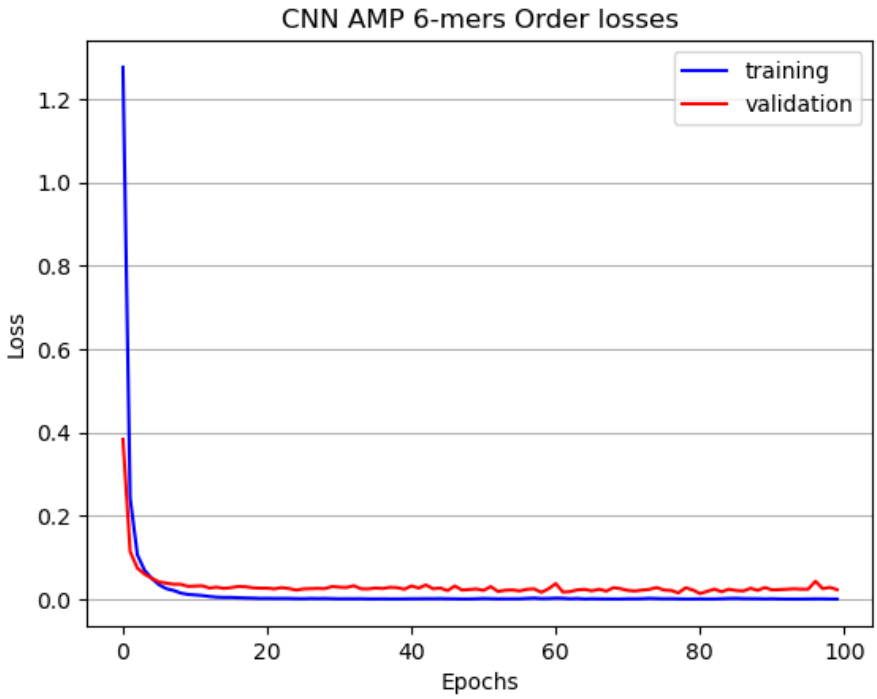

Figure 26: Training and validation loss curves for the CNN classifier tested with the AMP dataset and 6-mers at order taxonomic rank.

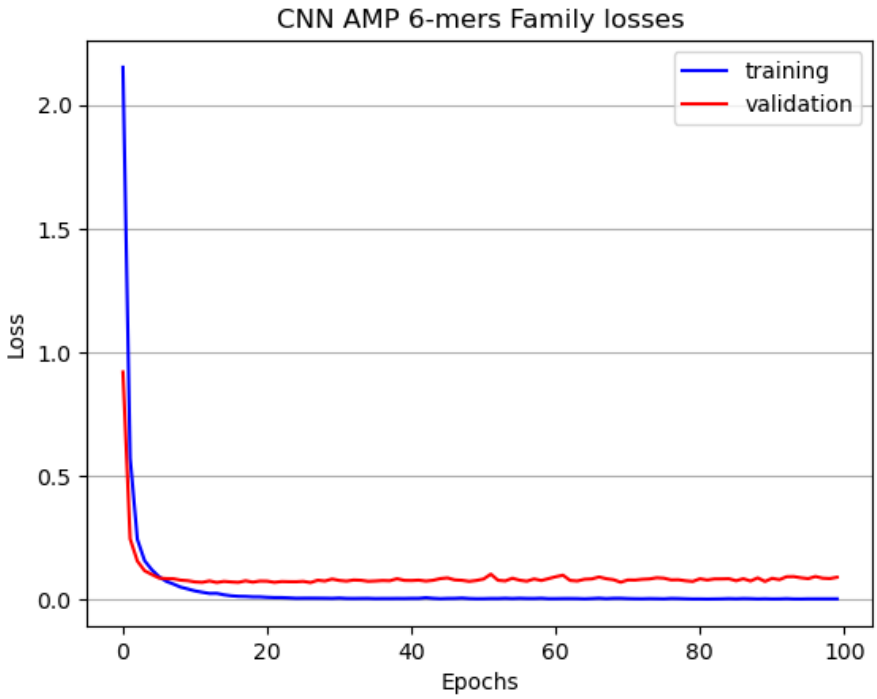

Figure 27: Training and validation loss curves for the CNN classifier tested with the AMP dataset and 6-mers at family taxonomic rank.

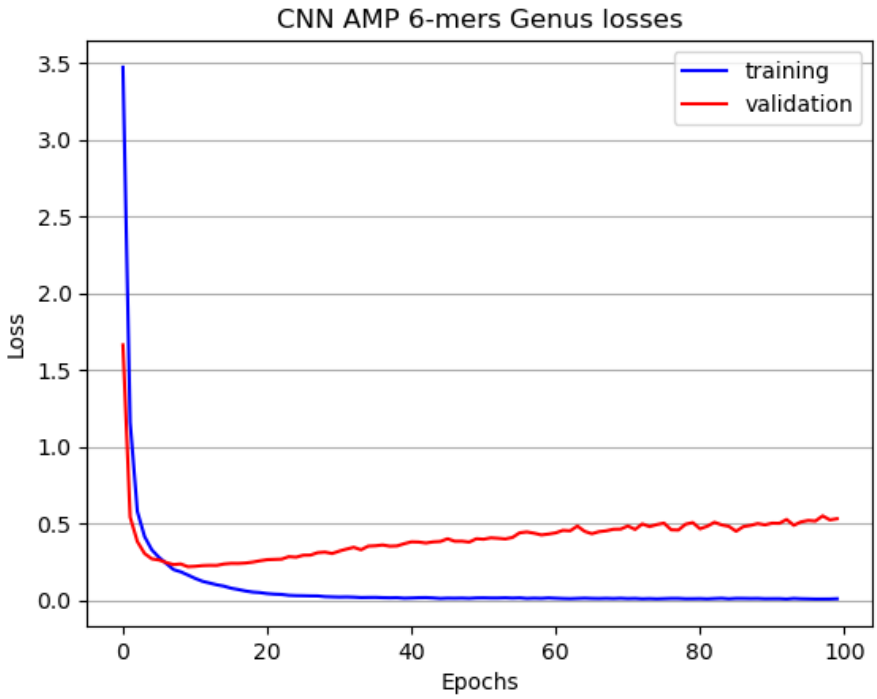

Figure 28: Training and validation loss curves for the CNN classifier tested with the AMP dataset and 6-mers at genus taxonomic rank.

3.1.6 Kernel size hyperparameter

A Convolutional Neural Network has many parameters which may be tuned. In this case, the kernel size of both convolutional layers used is changed, from 5 to 4 and 6, in order to detect if it is an essential parameter or the model is able to detect general information properly. The SG dataset with k-mer size equals 7 is the first that have been used, and no change is appreciated in the accuracy (Fig.29). For the AMP dataset, the k-mer size used has been 6 as the accuracy is similar with 6-mers and 7-mers. The result for the AMP does not bring any new information, as the model performance is still the same for different kernel size in the convolutional layers, as shown in Fig.30.

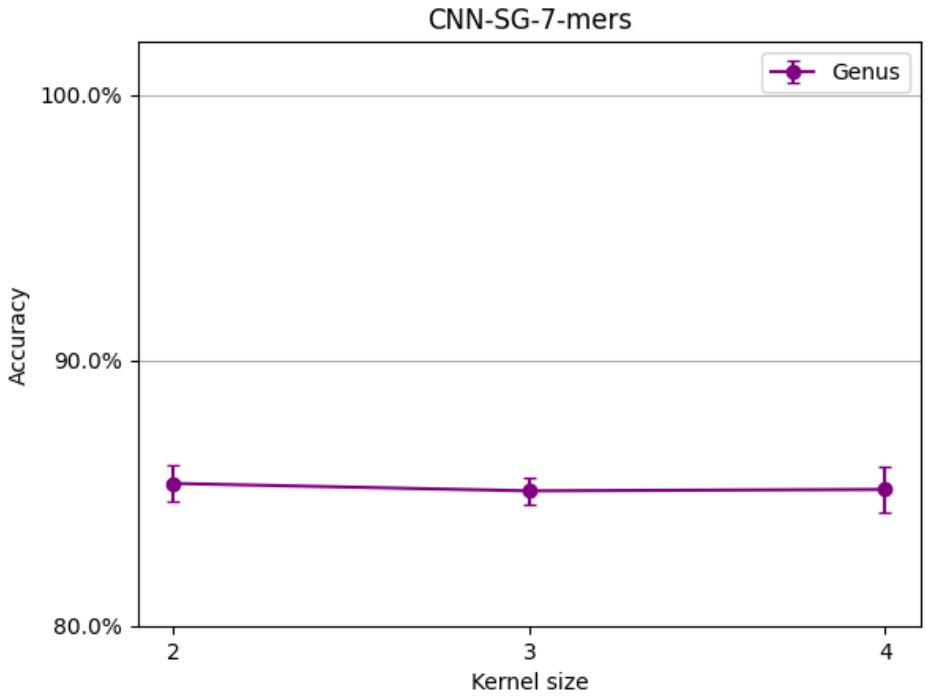

Figure 29: Accuracy of the CNN model with SG dataset at genus level with different kernel size in both convolutional layers.

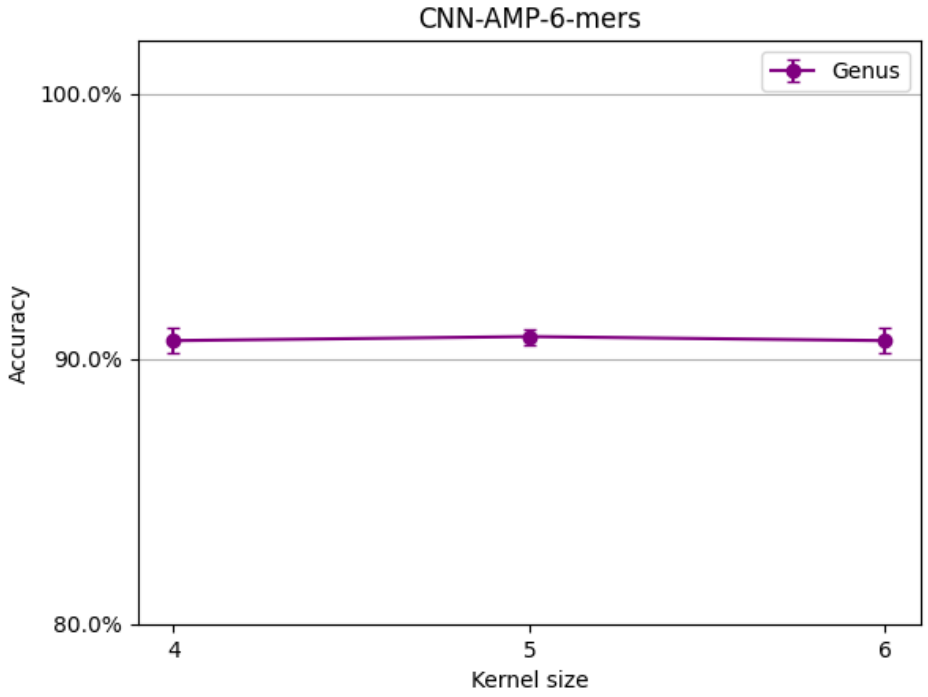

Figure 30: Accuracy of the CNN model with AMP dataset at genus level with different kernel size in both convolutional layers.

3.2 Deep Belief Network (DBN)

The DBN classifier is based on the combination of two Restricted Boltzmann Machines (RBM) with 256 neurons units in the hidden layer.

3.2.1 Accuracy

The accuracy of the DBN models trained with the SG and AMP datasets for different k-mer size and taxonomic ranks is represented in Fig.31 and Fig.32 respectively. The exact values can be seen in Table.A.11 and Table.A.12. Interestingly, the k-mer size has a huge impact when training with the SG dataset. In fact, the effect is so important that the accuracy does not converge for the largest k-mer size at genus level. In contrast, the accuracy apparently reach their maximum when training with the AMP dataset, since nearly no difference is seen when using 5-mers, 6-mers and 7-mers. This behaviour is quite similar to the one shown by the CNN.

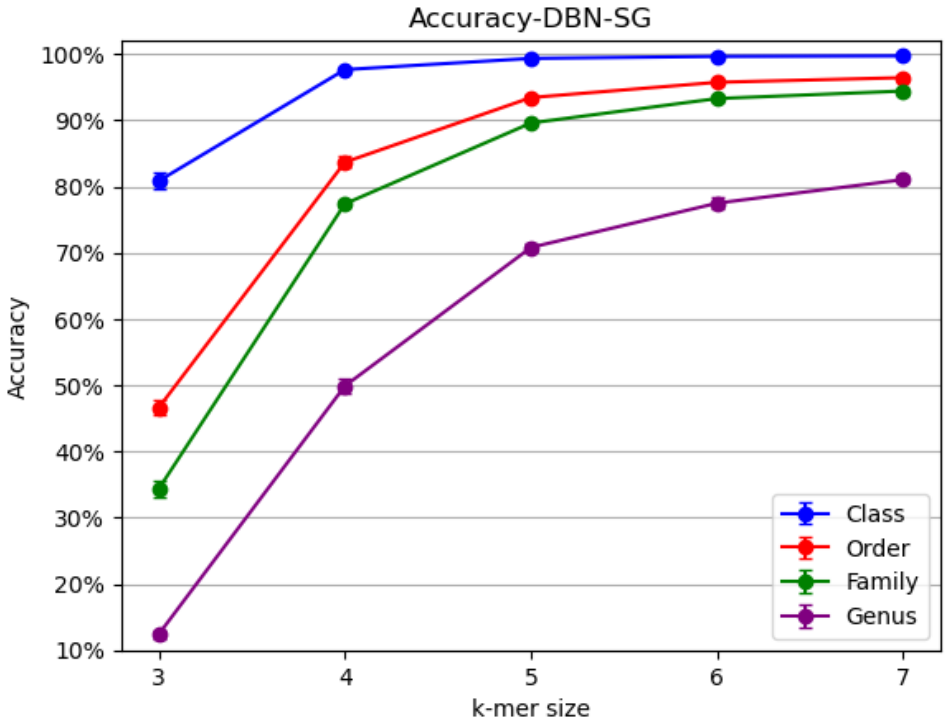

Figure 31: Accuracy of the DBN model with SG dataset.

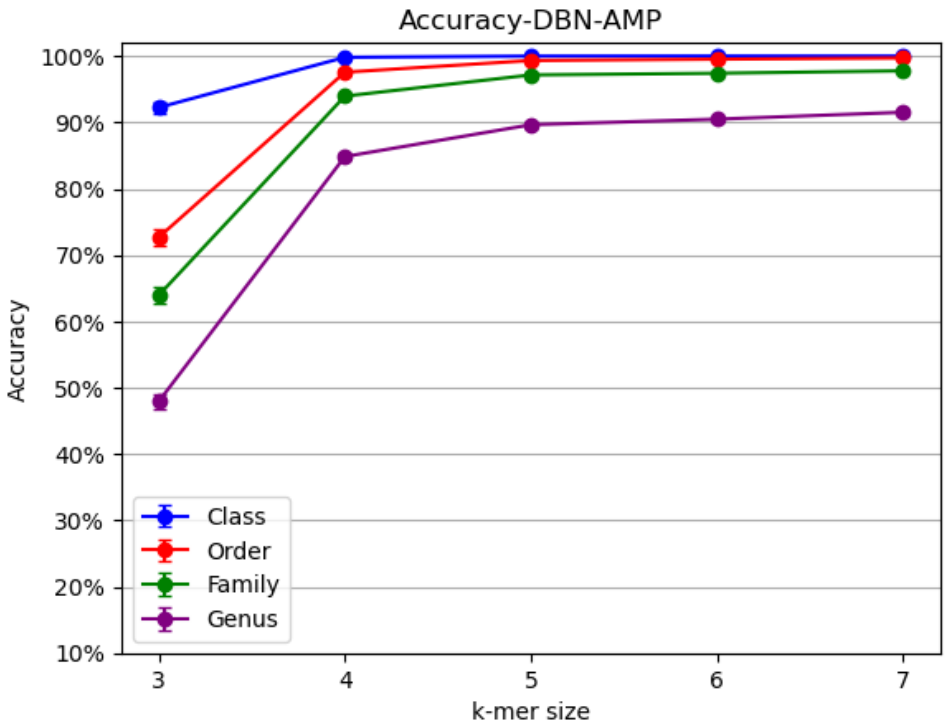

Figure 32: Accuracy of the DBN model with AMP dataset.

3.2.2 Computing time

Perhaps the most important limitation of the DBN classifier is the computing time. The training time is extremely high, taking around 4 hours for each fold, so a total of 40 hours for each model since 10 folds are used. The model takes so much time that no difference can be appreciated between training with the SG dataset or the AMP nor with the k-mer size, as can be seen in Fig.33 and Fig.34. The computing time is also high regardless of the taxonomic rank (class, order, family or genus). Thereby, it can be concluded that the DBN is unstable, in the sense that it is neither constant nor consistent, as far as training time is concerned. The exact values of the training time are presented in Table.A.13 and Table.A.14 for the SG and AMP datasets, respectively, while the inference time are in Table.A.15 and Table.A.16.

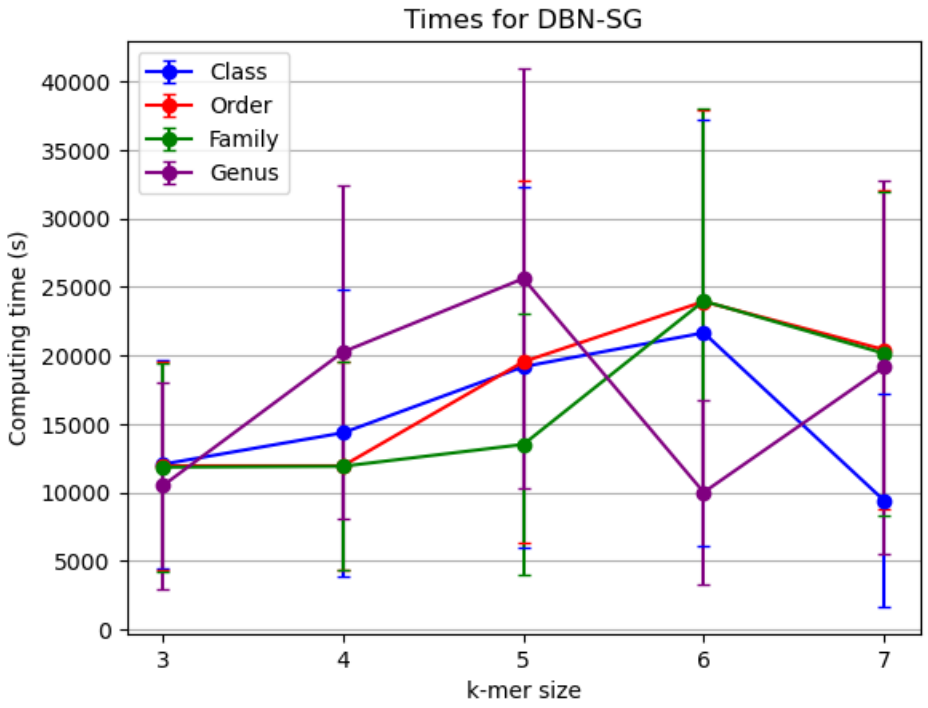

Figure 33: Computing time (s) for the DBN model and SG dataset vs the k-mers size.

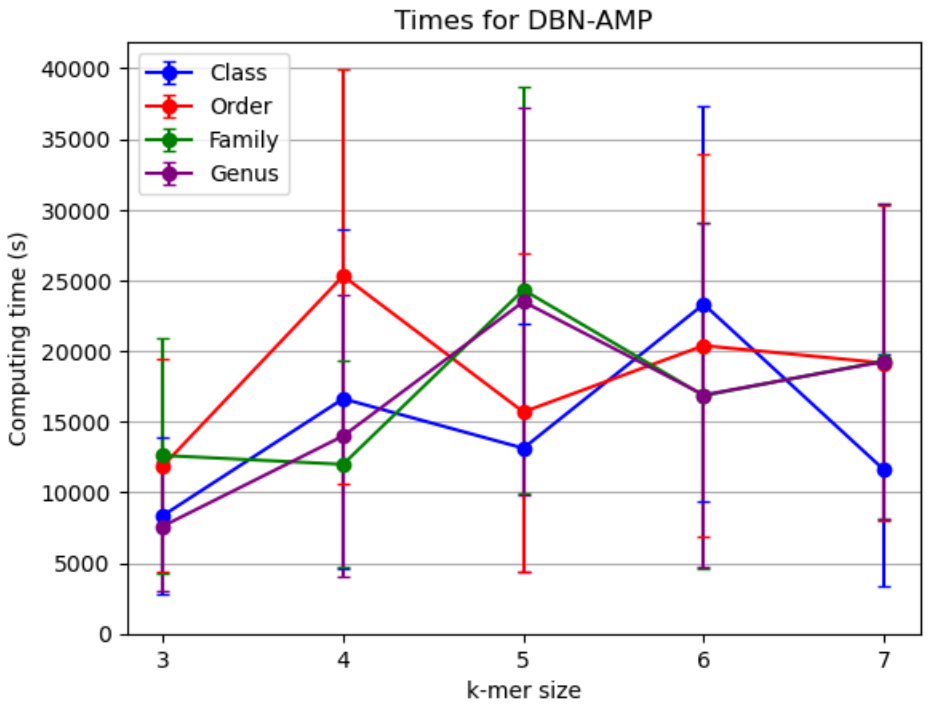

Figure 34: Computing time (s) for the DBN model and AMP dataset vs the k-mers size.

3.2.3 Other metrics

Regarding the precision, recall, F1 score and Area Under the Curve (AUC), significant difference are observed comparing the SG and AMP datasets. Fig.35 and Fig.36 represents the precision for the DBN models trained with SG and AMP. Analogously, for the recall, the results are shown in Fig.37 and Fig.38. Then, the F1 score is presented in Fig.39 and Fig.40, while the AUC metric is in Fig.41 and Fig.42. As happened with the CNN classifier, the models trained with AMP have a slightly better performance than the ones trained with SG, basically because it reaches convergence before and probably due to overfitting as well. The specific value for precision, recall, F1 score and AUC can be seen in Appendix A in Table.A.17, Table.A.18, Table.A.19 and Table.A.20.

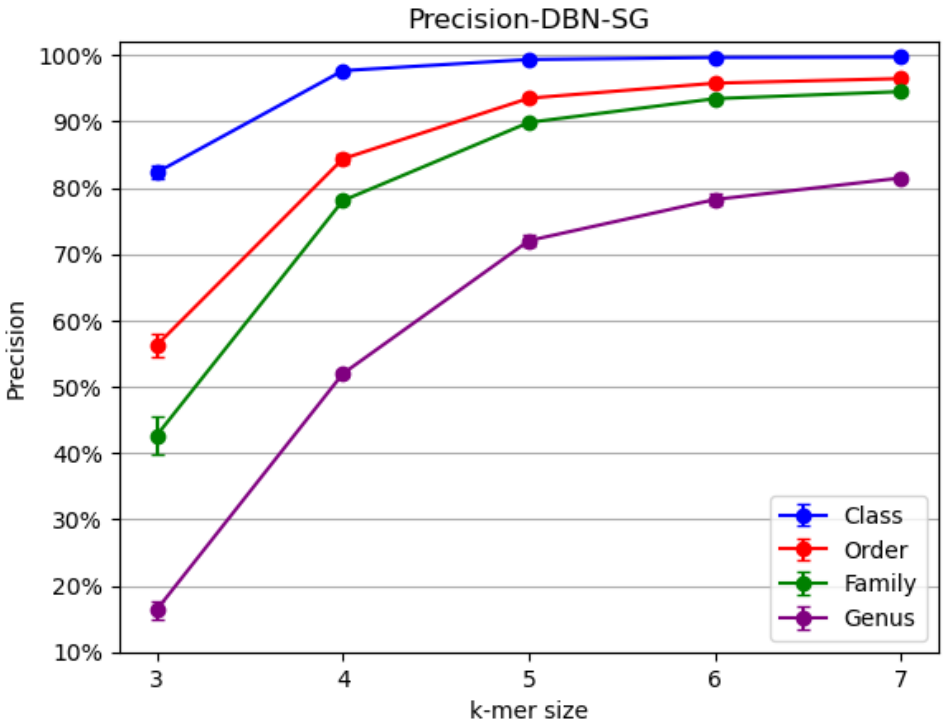

Figure 35: Precision of the DBN model with SG dataset.

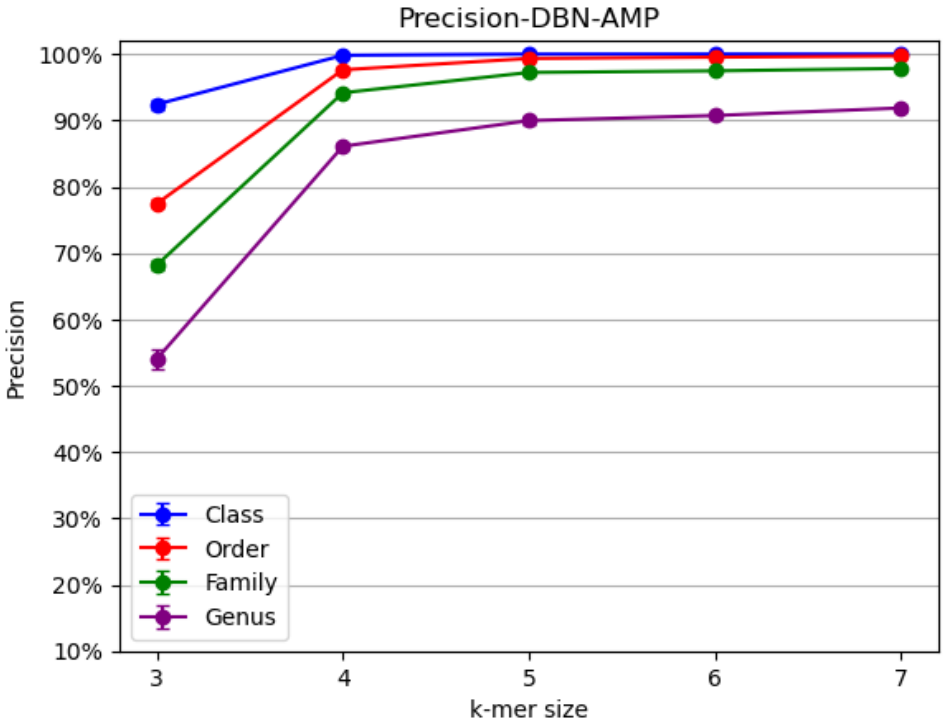

Figure 36: Precision of the DBN model with AMP dataset.

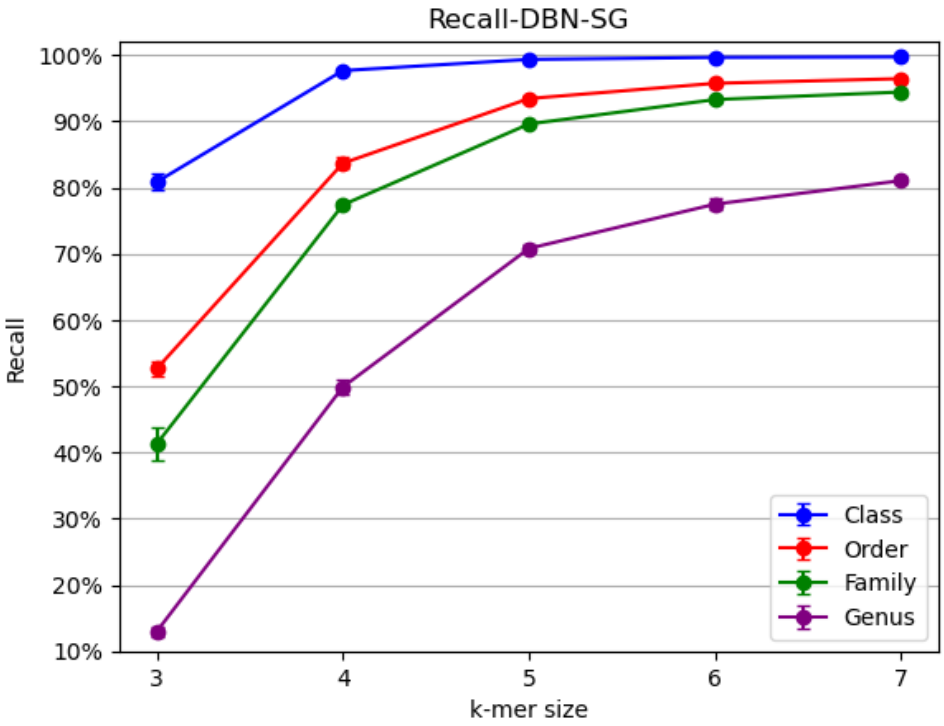

Figure 37: Recall of the DBN model with SG dataset.

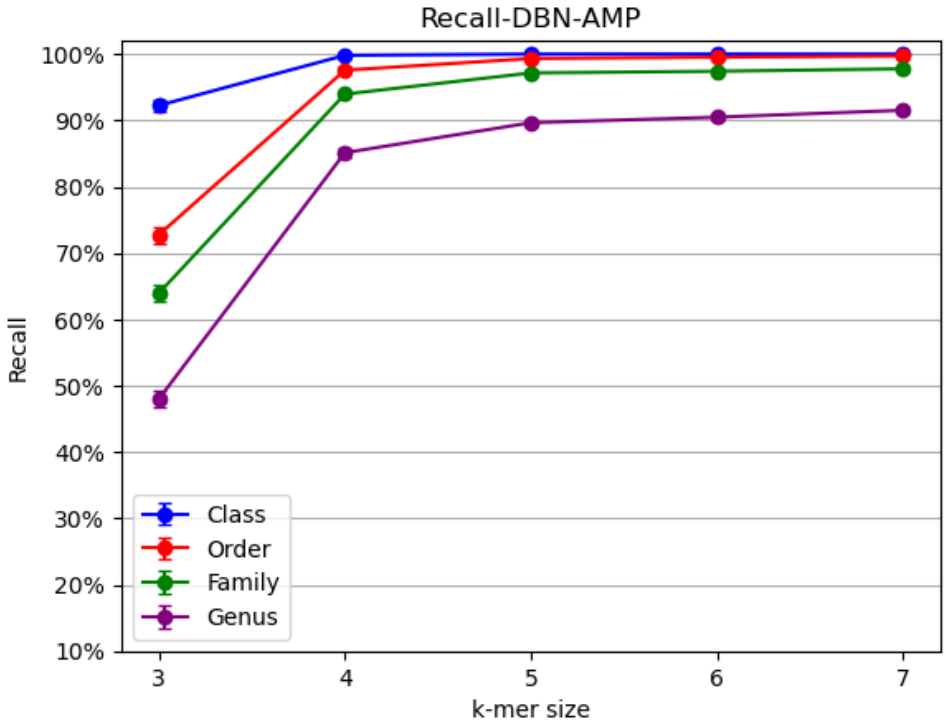

Figure 38: Recall of the DBN model with AMP dataset.

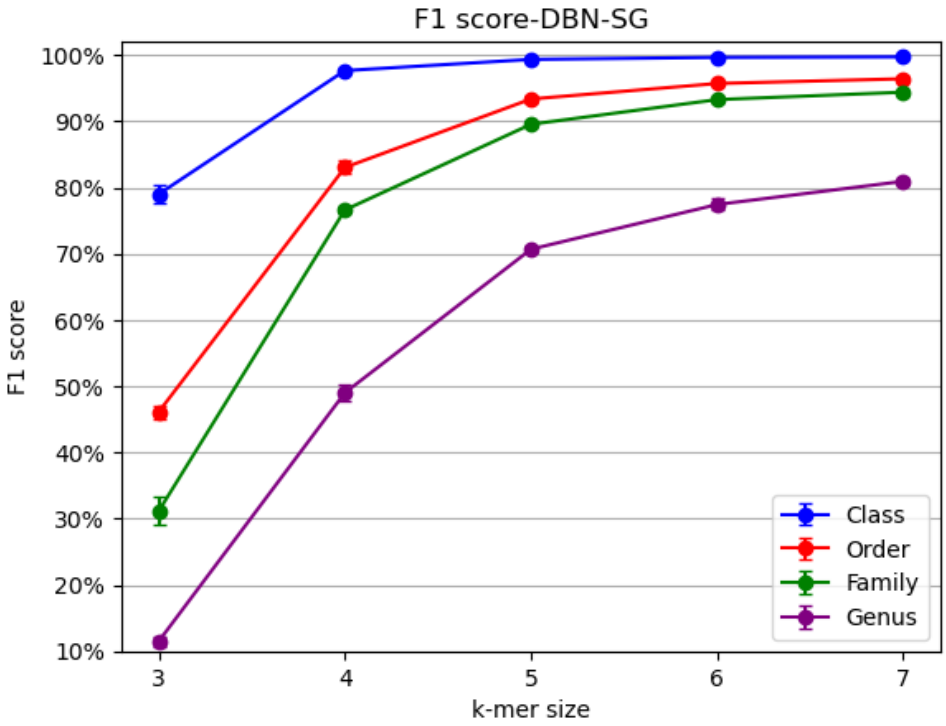

Figure 39: F1 score of the DBN model with SG dataset.

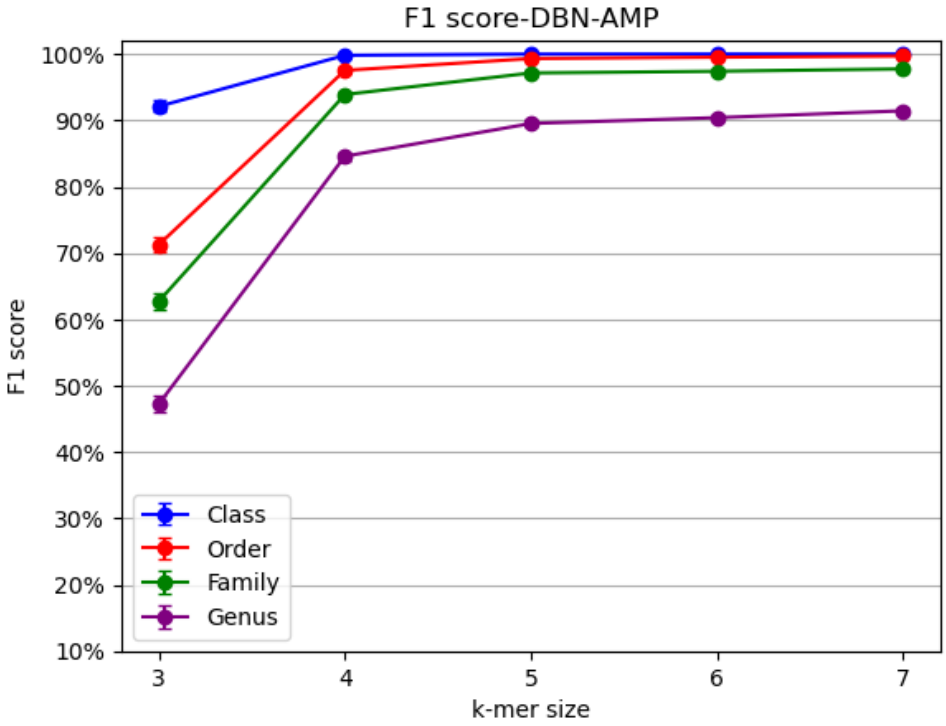

Figure 40: F1 score of the DBN model with AMP dataset.

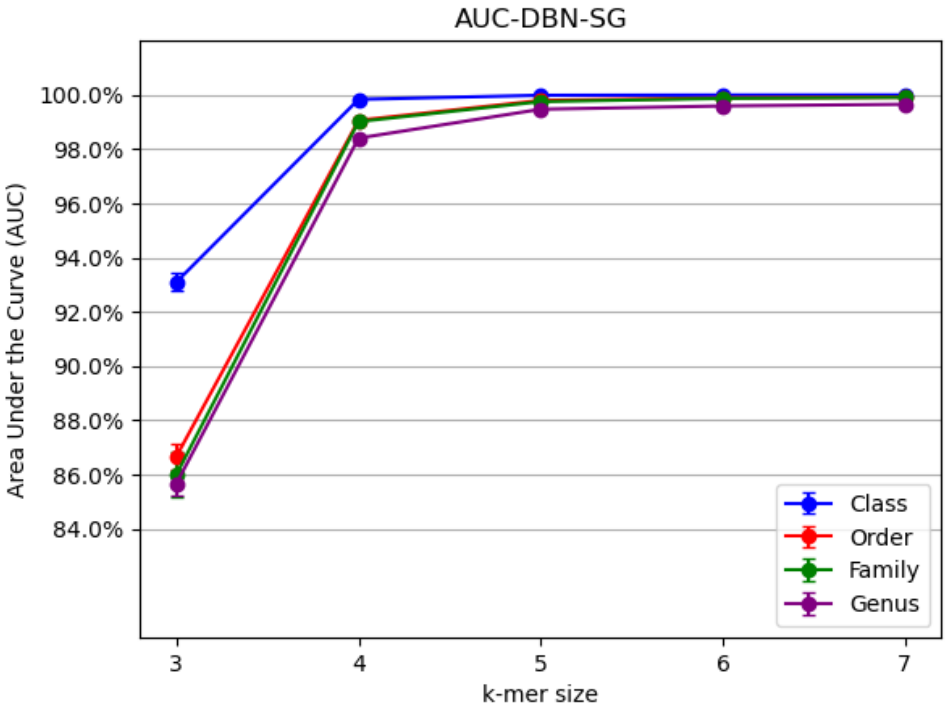

Figure 41: Area Under the Curve (AUC) of the DBN model with SG dataset.

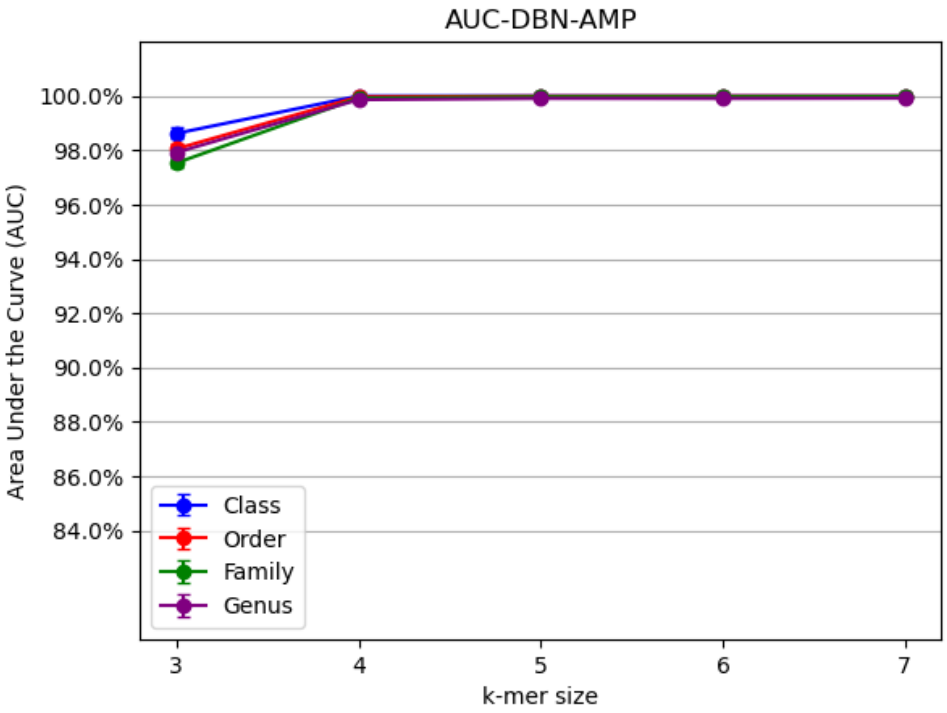

Figure 42: Area Under the Curve (AUC) of the DBN model with AMP dataset.

3.2.4 Confusion matrix

As in the previous classifier, the model trained with the SG dataset and 7-mers has been selected as representative to show some confusion matrices. Randomly, one of the folds has been used. At the taxonomic level of genus there are some important errors regarding some groups, the 10 groups with more problems are represented in Fig.43. At class level, as expected, there is nearly no misclassification observed, as it can be seen in Fig.A.4, since is the easiest taxonomic rank to distinguish and may be suffering overfitting. Then, at the order and family level there are some misclassifications as it can be observed in Fig.A.5 and Fig.A.6 respectively.

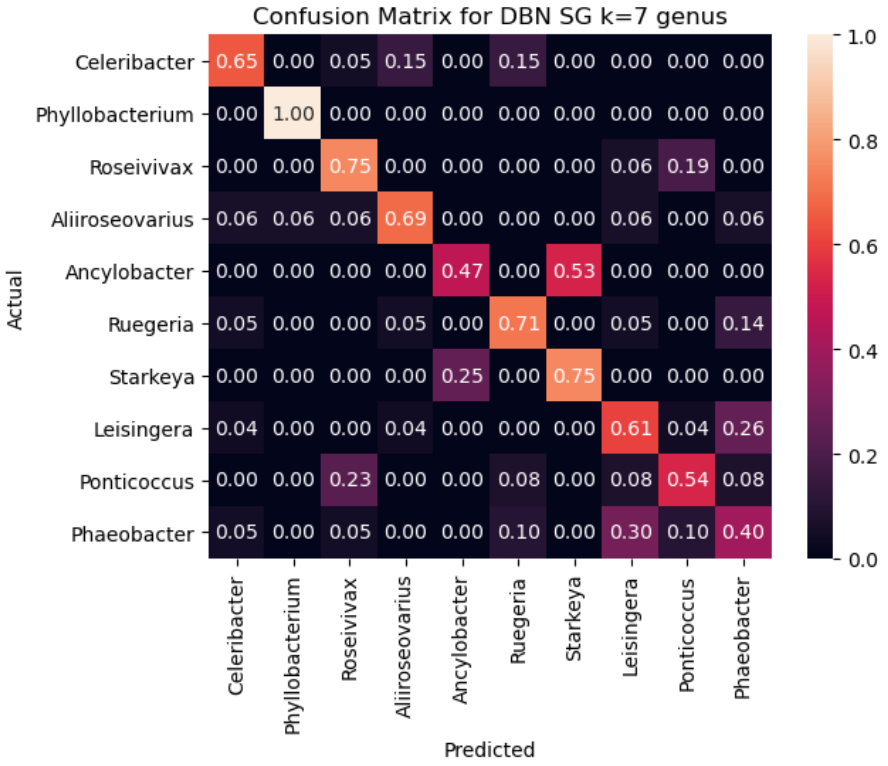

Figure 43: Confusion matrix of the DBN model at the genus level, trained with the SG dataset and 7-mers. Just the genus with more misclassifications are shown.

3.2.5 Loss curves

The training of the DBN model is quite time-consuming. Understanding the training and validation loss may be interesting to know how well the model is learning. As in the previous algorithm, a fold of the model trained with the SG dataset and 7-mers and another of the model trained with the AMP dataset and 6-mers are used to show the loss versus the epochs. For the model trained with SG, the loss curves are shown in Fig.44, Fig.45, Fig.46 and Fig.47 for the class, order, family and genus taxonomic rank respectively. The model seems to be learning the parameters correctly, however, specifically at genus level, the validation loss is significantly higher than the training loss which indicates that the classifier is not able to deal adequately with unknown data. In other words, the model is suffering overfitting. For the model trained with AMP dataset, the curves can be seen in Fig.48, Fig.49, Fig.50 and Fig.51. In this case the model present a lower lost than before at genus level, although the

overfitting is still there, in fact is easily observable since the validation loss increases in high epochs. At class, order and family level, the model seems to be learning successfully, but it probably suffers overfitting as well. Since the data is easier to distinguish than at genus level, the impact of the overfitting may be less notable.

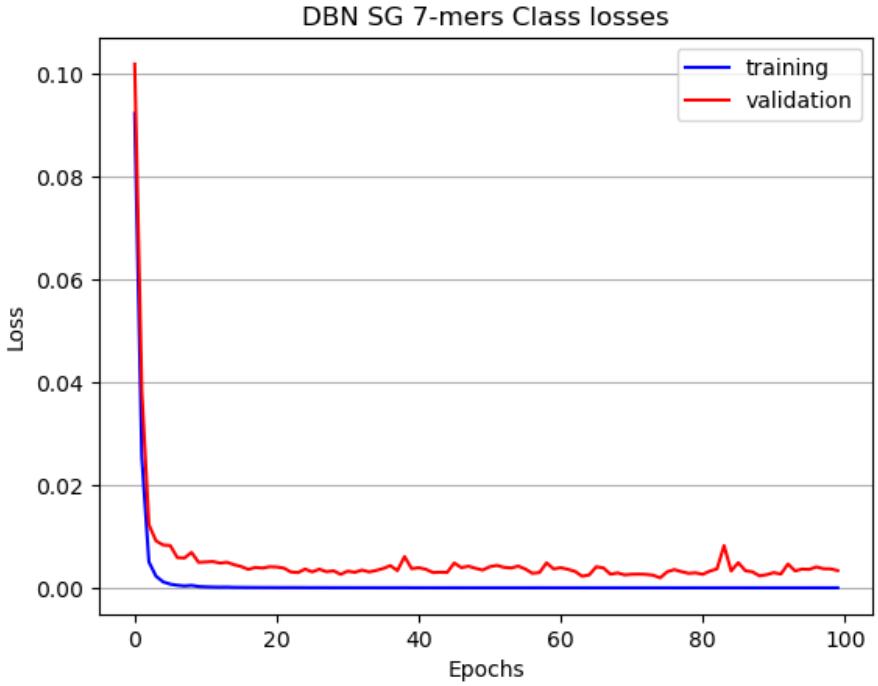

Figure 44: Training and validation loss curves for the DBN classifier tested with the SG dataset and 7-mers at class taxonomic rank.

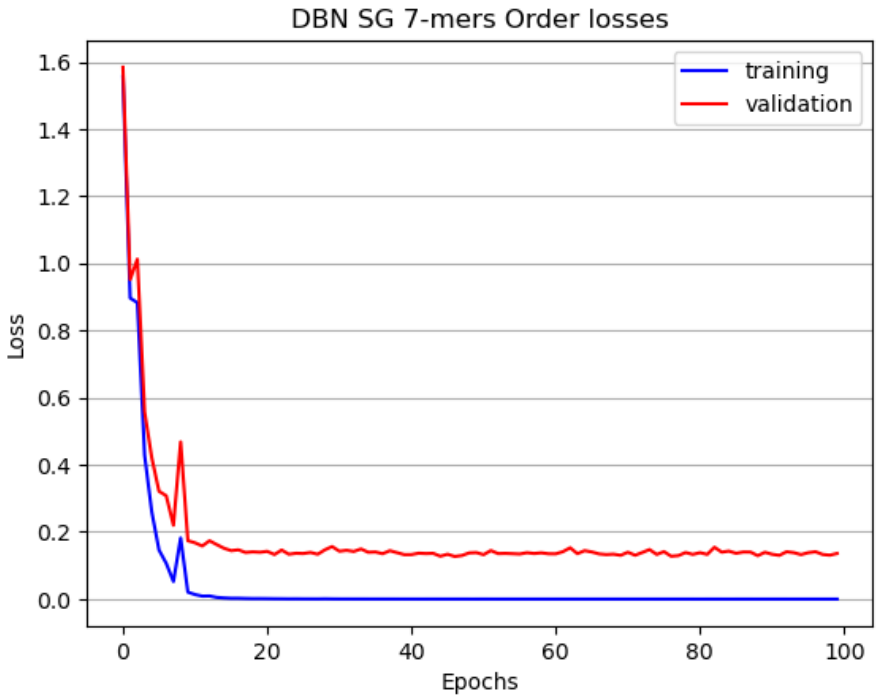

Figure 45: Training and validation loss curves for the DBN classifier tested with the SG dataset and 7-mers at order taxonomic rank.

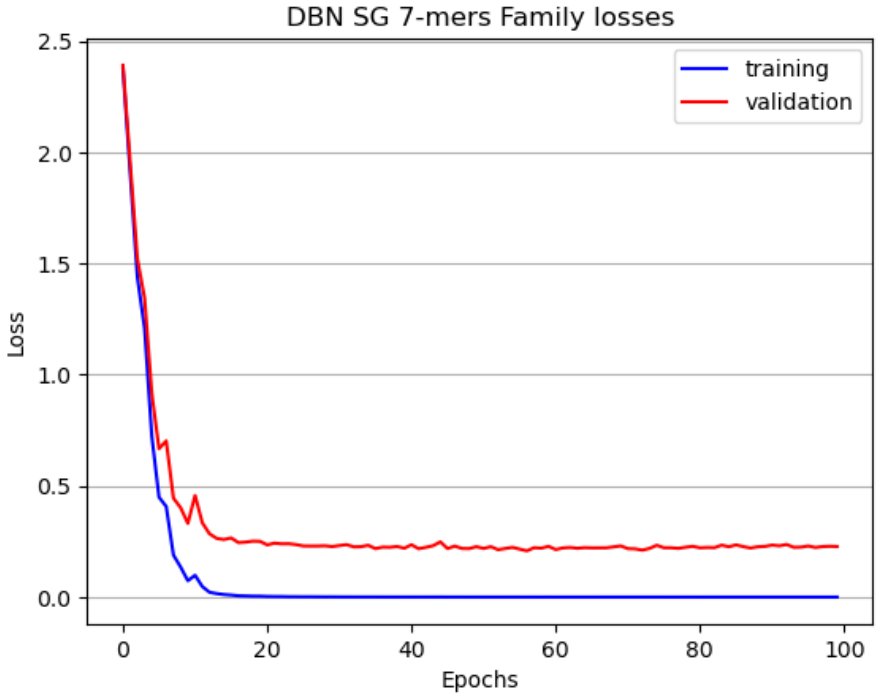

Figure 46: Training and validation loss curves for the DBN classifier tested with the SG dataset and 7-mers at family taxonomic rank.

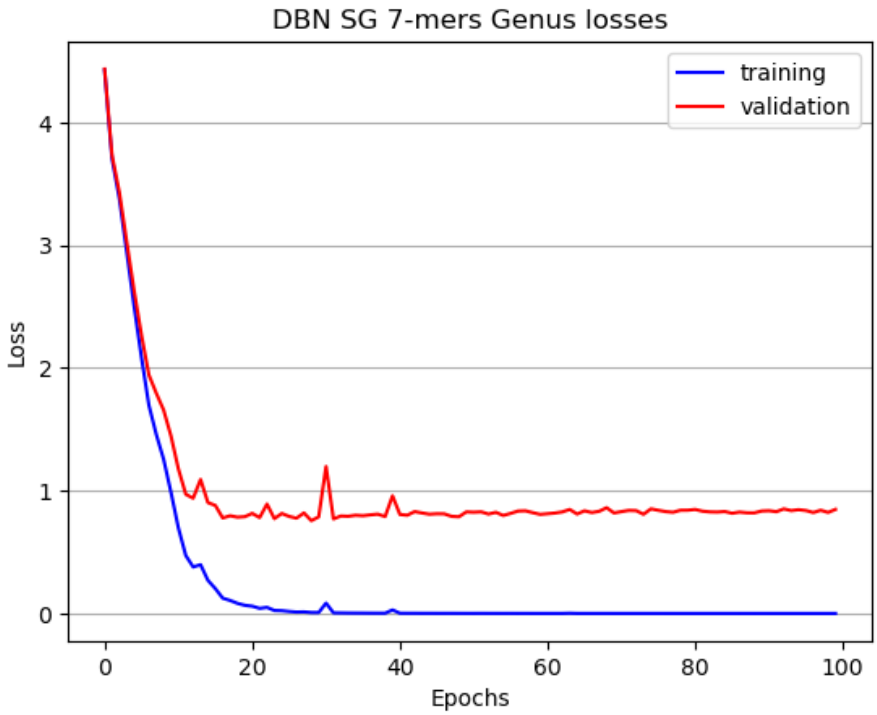

Figure 47: Training and validation loss curves for the DBN classifier tested with the SG dataset and 7-mers at genus taxonomic rank.

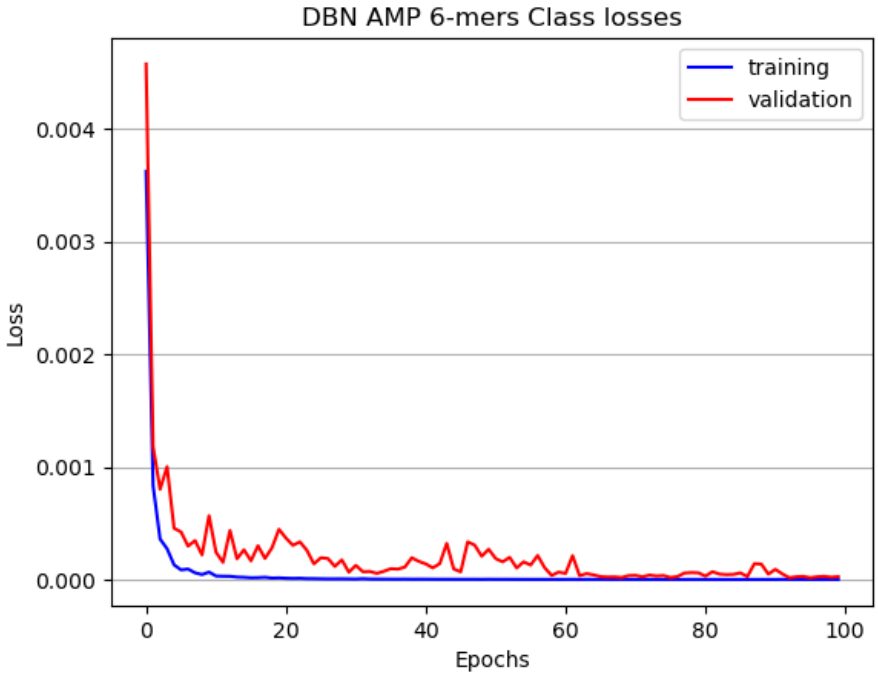

Figure 48: Training and validation loss curves for the DBN classifier tested with the AMP dataset and 6-mers at class taxonomic rank.

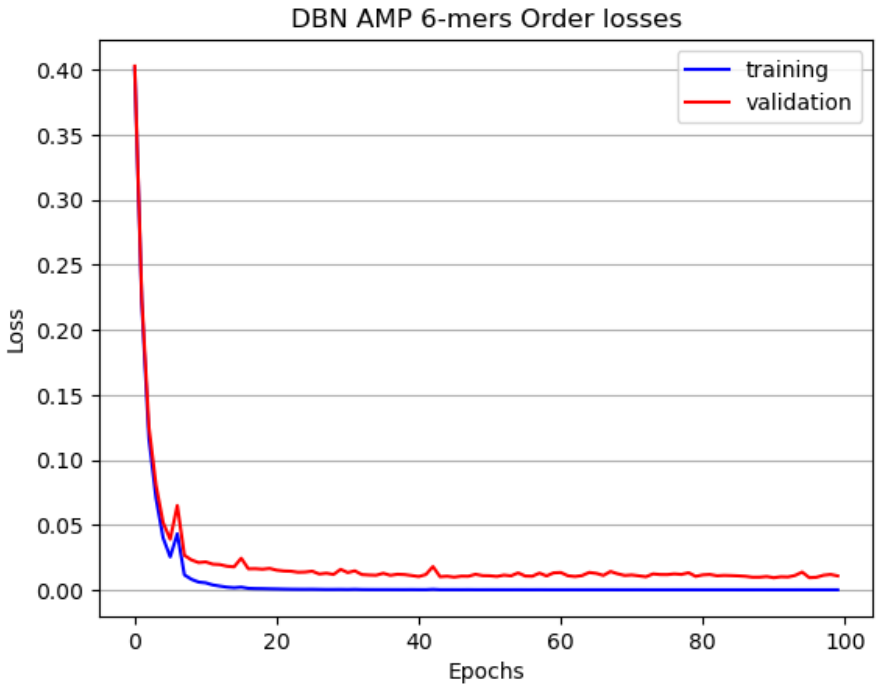

Figure 49: Training and validation loss curves for the DBN classifier tested with the AMP dataset and 6-mers at order taxonomic rank.

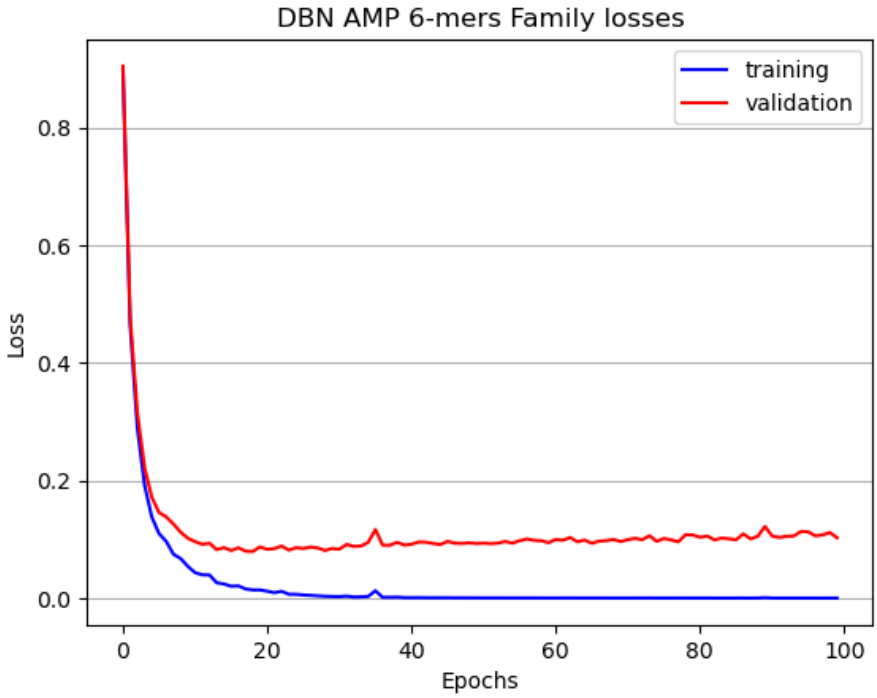

Figure 50: Training and validation loss curves for the DBN classifier tested with the AMP dataset and 6-mers at family taxonomic rank.

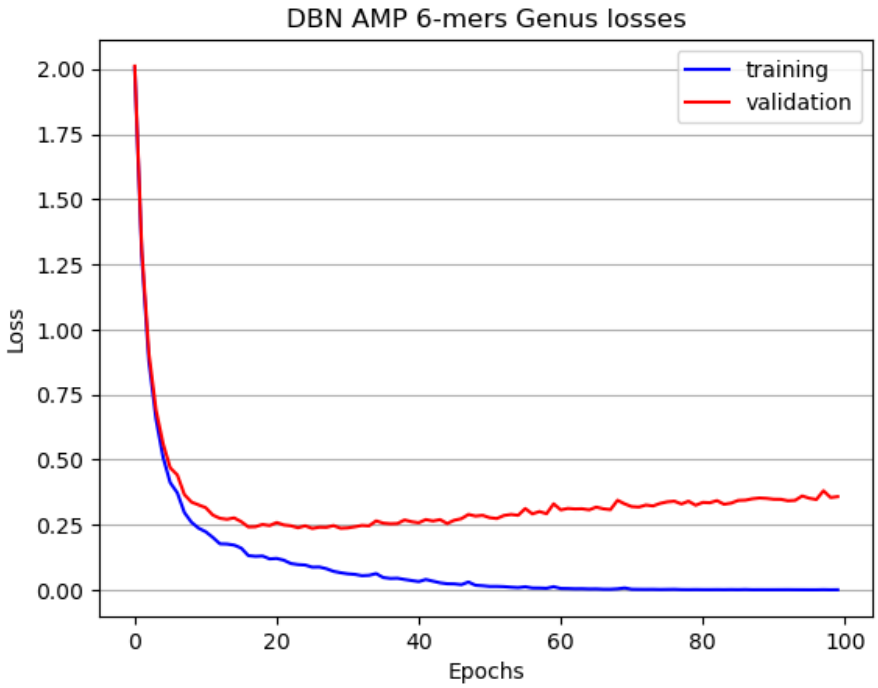

Figure 51: Training and validation loss curves for the DBN classifier tested with the AMP dataset and 6-mers at genus taxonomic rank.

3.3 XGBoost

The XGBoost model implemented has the default values. This means a maximum tree depth of 3 and a learning rate of 0.1.

3.3.1 Accuracy

The behaviour of this model regarding the k-mer size and the taxonomy rank is similar to the previous models. Using the SG dataset, as it can be observed in Fig.52, the improvement of the accuracy is significant when increasing the k-mer size. In contrast, using the AMP, shown in Fig.53, such sharp rise is not present since with 5-mers the accuracy is notably high. Again, the difference between the accuracy obtained for the genus level is clearly lower than for the other levels, regardless of the k-mer size. The accuracy values can be read in detail in Table.A.21 for the SG dataset and in Table.A.22 for the AMP dataset.

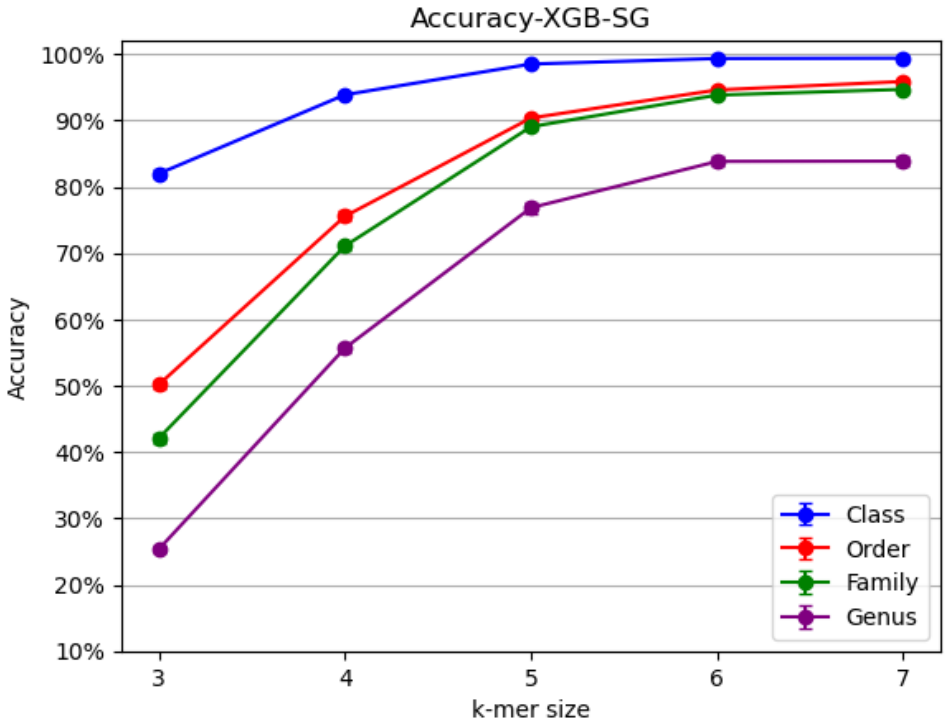

Figure 52: Accuracy of the XGBoost model with SG dataset.

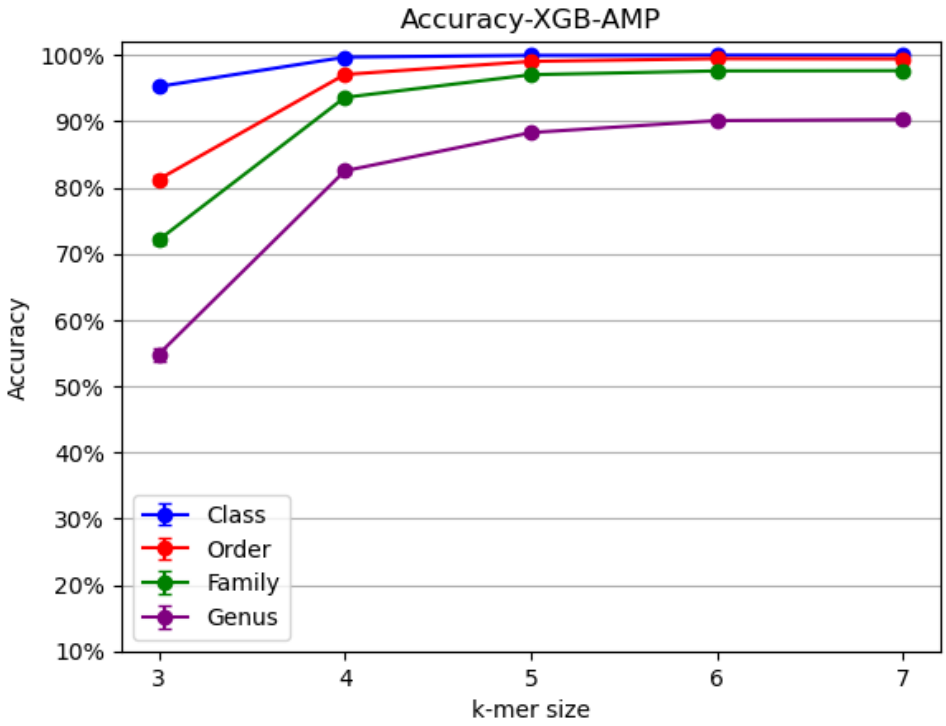

Figure 53: Accuracy of the XGBoost model with AMP dataset.

3.3.2 Computing time

Then, looking at the computing time, the training time exponentially increases with the k-mer size. This huge impact of the k-mer size implies a limitation when selecting the best k-mer size for this model. The algorithm training process is nearly instant when using 3-mers, 4-mers and 5-mers regardless of the taxonomic rank. However, using 7-mers, the training step takes around 7 hours in total at genus level, 4 at family level, 2 at order level and is still being nearly immediate at class level. For the SG dataset, the exact computing time can be seen in Table.A.23. For the AMP dataset (Table.A.24), the tendency is the same, although the times are systematically lower than with the SG dataset. In any case, the computing time of XGBoost is quite low compared with CNN and DBN algorithms when using a small k-mer size. The exponential tendency may be observed in the graphs presented in Fig.54 and Fig.55. In terms of the inference time, the model is tremendously quick, taking hundredths of a second, as Table.A.25 and Table.A.26 show.

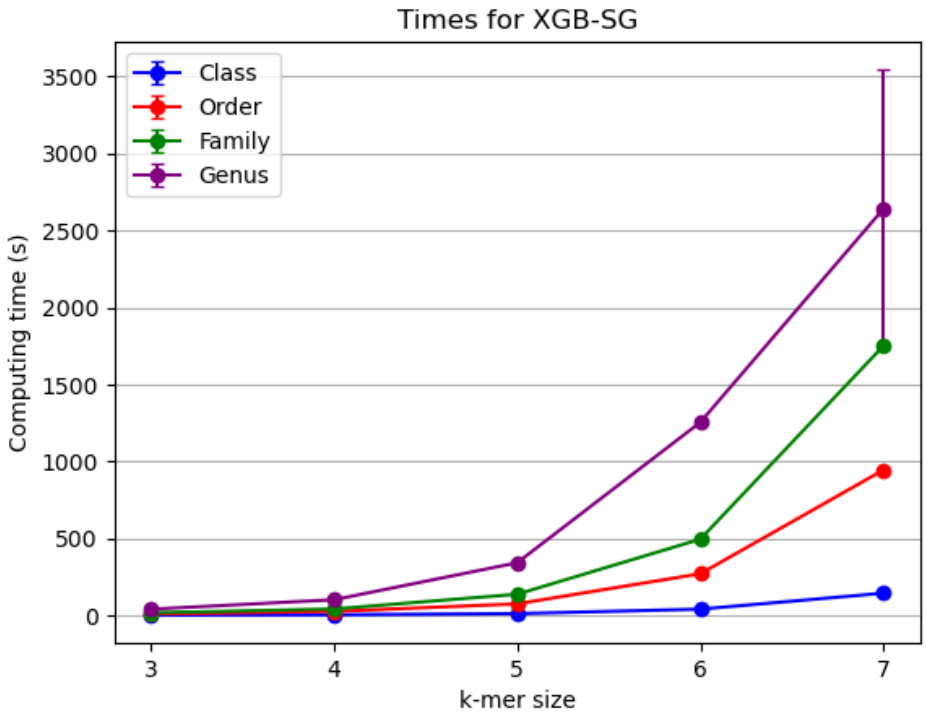

Figure 54: Computing time (s) for the XGBoost model and SG dataset vs the k-mers size.

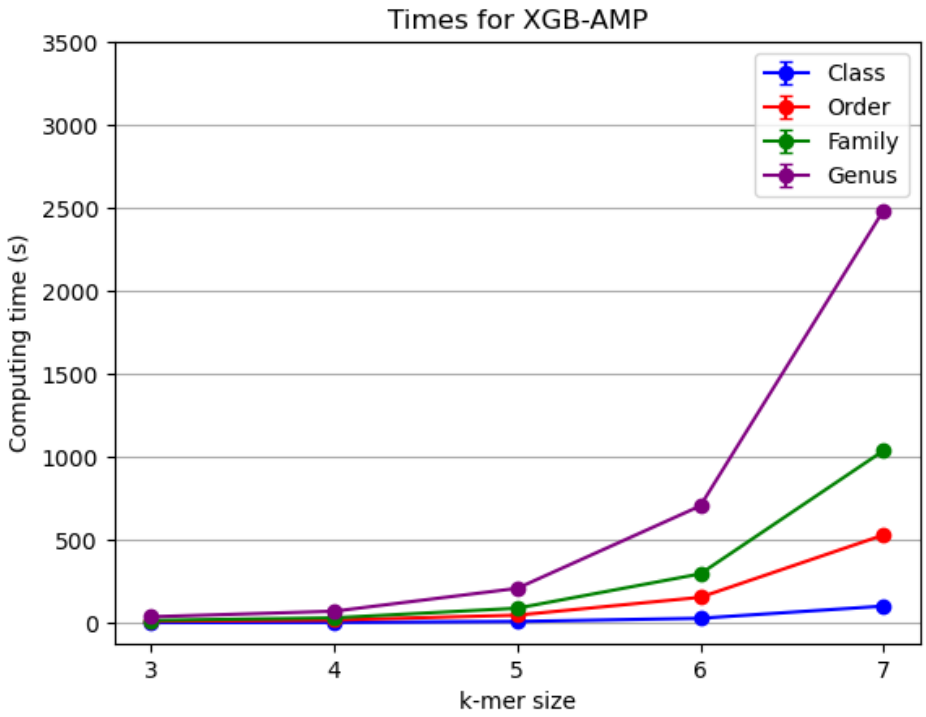

Figure 55: Computing time (s) for the XGBoost model and AMP dataset vs the k-mers size.

3.3.3 Other metrics

Now, taking into the account the precision, recall, F1 score and Area Under the Curve (AUC) metrics. The corresponding results showing the precision metric for models with different k-mer sizes trained with the SG dataset are in Fig.56 and trained with the AMP dataset in Fig.57. In the case of the recall metric are presented in Fig.58 and Fig.59. As for the F1 score, the results are shown in Fig.60 and Fig.61 and the AUC value evolution is represented in Fig.62 and Fig.63. All metrics indicate that a higher k-mer size is better than a low one, since the scores achieved are greater. However, it may be highlighted that for both dataset (SG and AMP), there is no clear difference between 6-mers and 7-mers strategy since the values in the metrics stabilise. This contrast with the CNN and DBN models, in which only the model with AMP achieved convergence in regard to the k-mer size influence. The values of precision, recall, F1 score and AUC metrics can be read in Table.A.27, Table.A.28, Table.A.29 and Table.A.30.

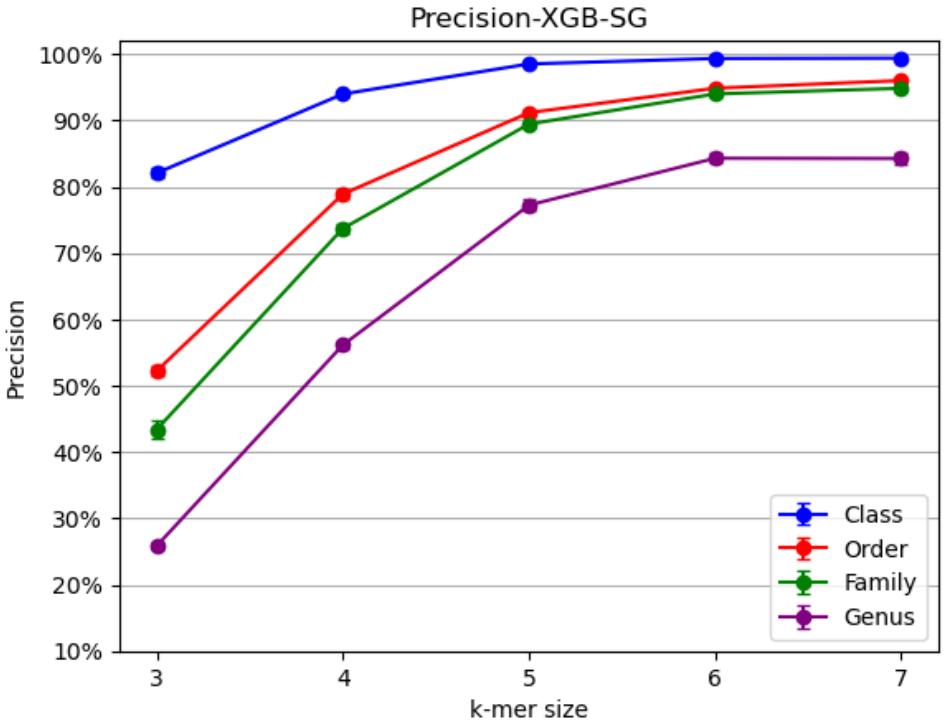

Figure 56: Precision of the XGBoost model with SG dataset.

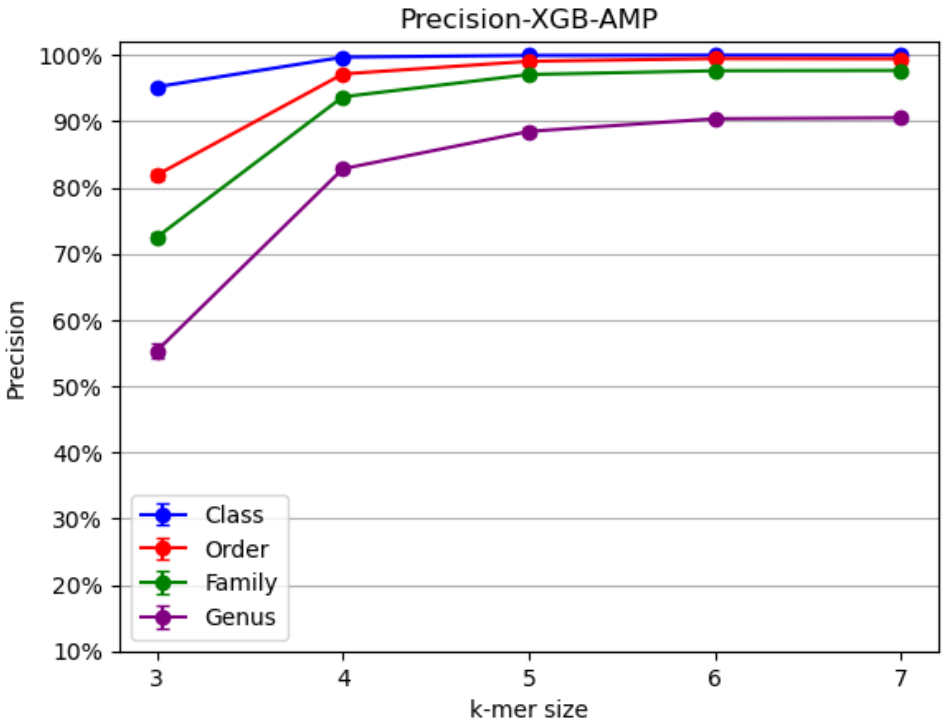

Figure 57: Precision of the XGBoost model with AMP dataset.

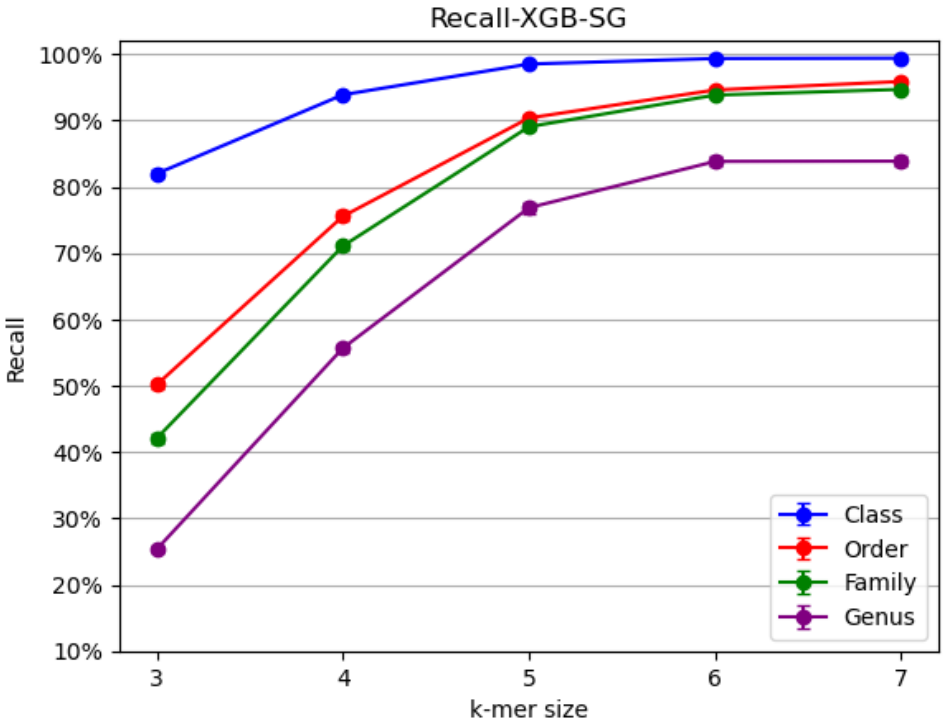

Figure 58: Recall of the XGBoost model with SG dataset.

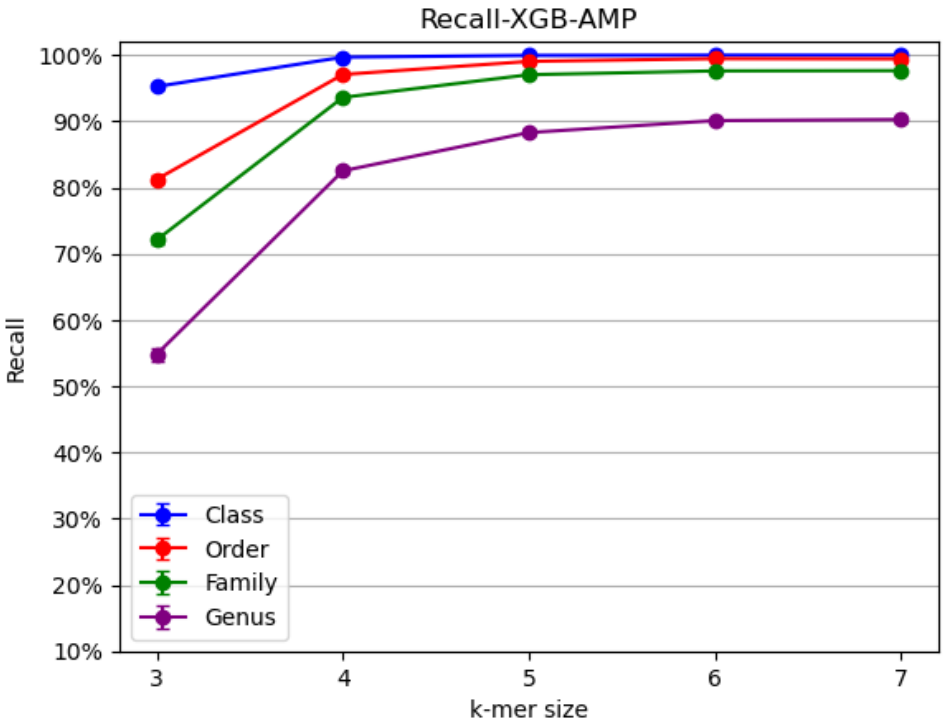

Figure 59: Recall of the XGBoost model with AMP dataset.

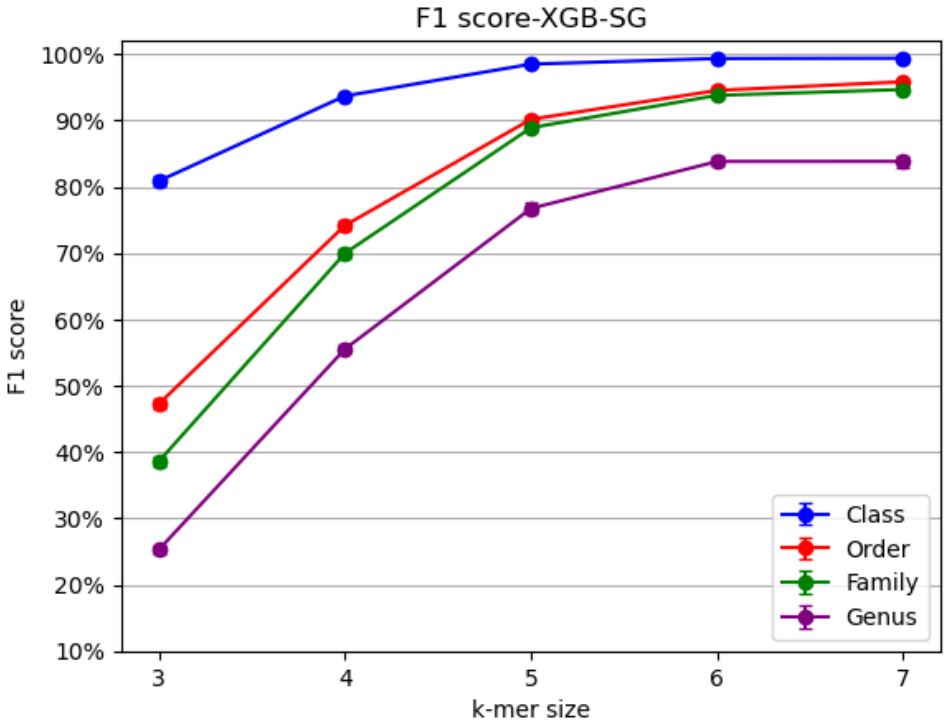

Figure 60: F1 score of the XGBoost model with SG dataset.

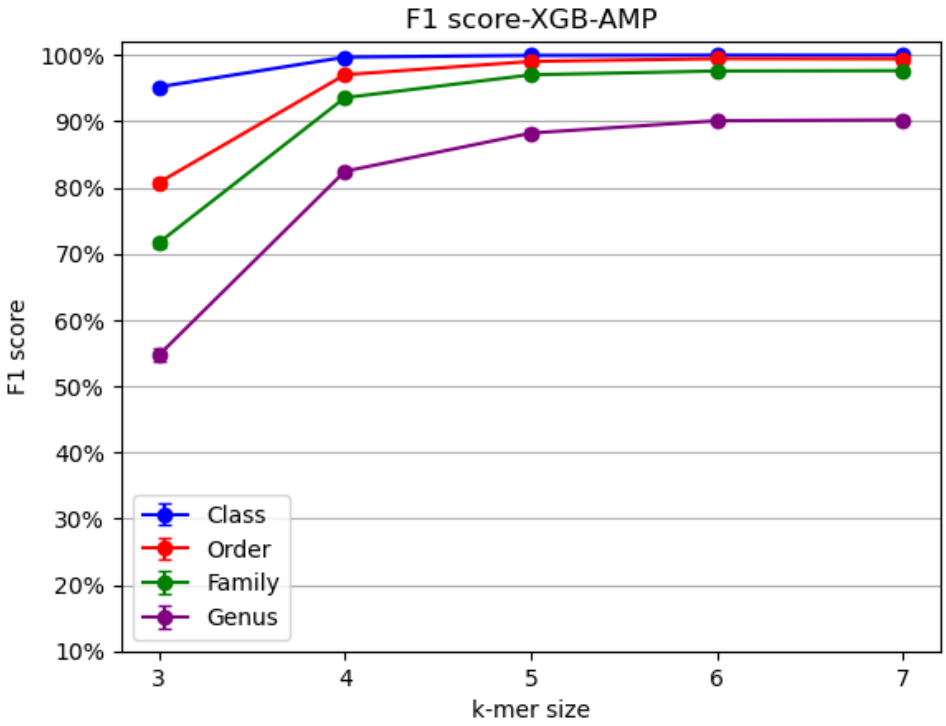

Figure 61: F1 score of the XGBoost model with AMP dataset.

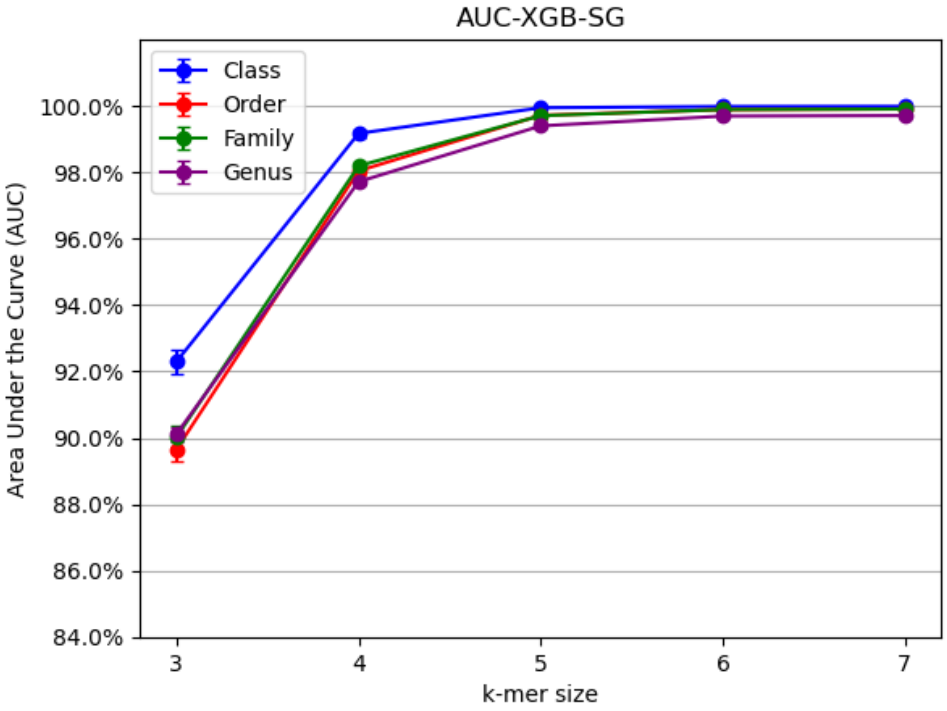

Figure 62: Area Under the Curve (AUC) of the XGBoost model with SG dataset.

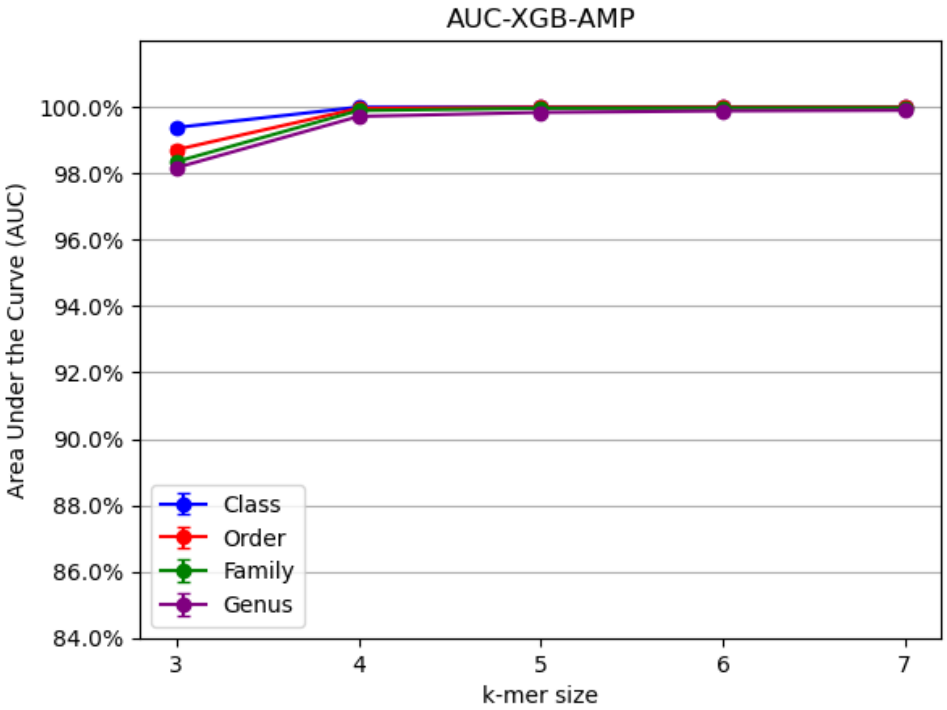

Figure 63: Area Under the Curve (AUC) of the XGBoost model with AMP dataset.

3.3.4 Confusion matrix

As in the previous classifiers, a fold of the model trained with the SG dataset and 7-mers is selected to show confusion matrices at different taxonomic levels to compare it with other models. The most critical genus levels are mostly the same as in the CNN and DBN classifiers trained with the same dataset (SG) and 7-mers representation. Fig.64 shows the 10 most interesting groups. Furthermore, the confusion matrix at class order is shown in Fig.A.7, at order level in Fig.A.8 and at family level in Fig.A.9.

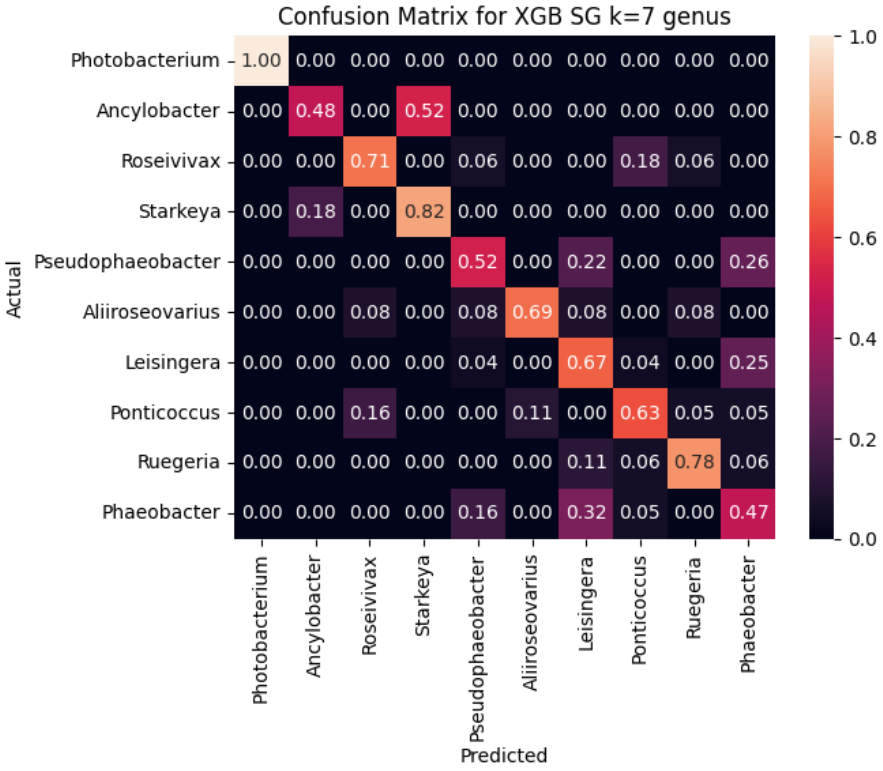

Figure 64: Confusion matrix of the XGBoost model at the genus level, trained with the SG dataset and 7-mers. Just the genus with more misclassifications are shown.

3.3.5 Loss curves

Regarding the training process of the XGBoost model, the training and validation loss behaviour are compared. As in the previous algorithms, the model trained with the SG dataset and 7-mers and the one trained with the AMP dataset and 6-mers

are used to analyse the process. Only a fold for each of them is used. For the first, the loss curves are shown in Fig.65, Fig.66, Fig.67 and Fig.68. The model seems to be learning the parameters correctly for all taxonomic levels without suffering significant problems, although a slight underfitting is observed. For the second model, trained with the AMP dataset, the curves can be seen in Fig.69, Fig.70, Fig.71 and Fig.72. In those graphics, no significant learning problems are observed, although the validation loss is always clearly higher than the training one, which may indicate overfitting. It is also interesting to point out that less than 100 epochs are enough to learn the structure of the data. In general, it may be highlighted that the loss curves for the XGBoost classifier are smooth, since no noise appears in any of the plotted losses. Interestingly, it is also noticeable that with the AMP dataset, the classifier learns with fewer epochs than with the SG dataset.

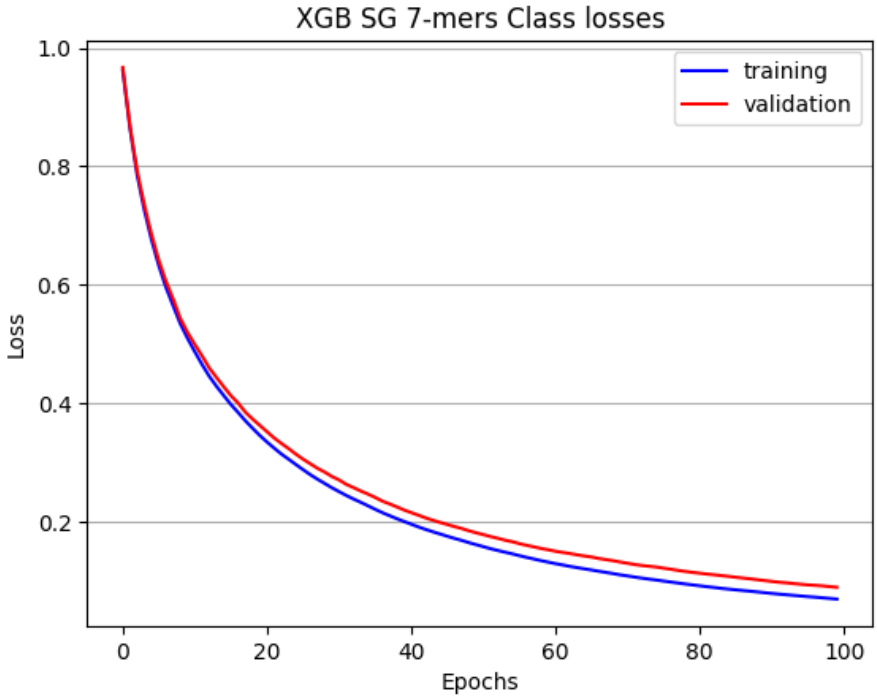

Figure 65: Training and validation loss curves for the XGBoost classifier tested with the SG dataset and 7-mers at class taxonomic rank.

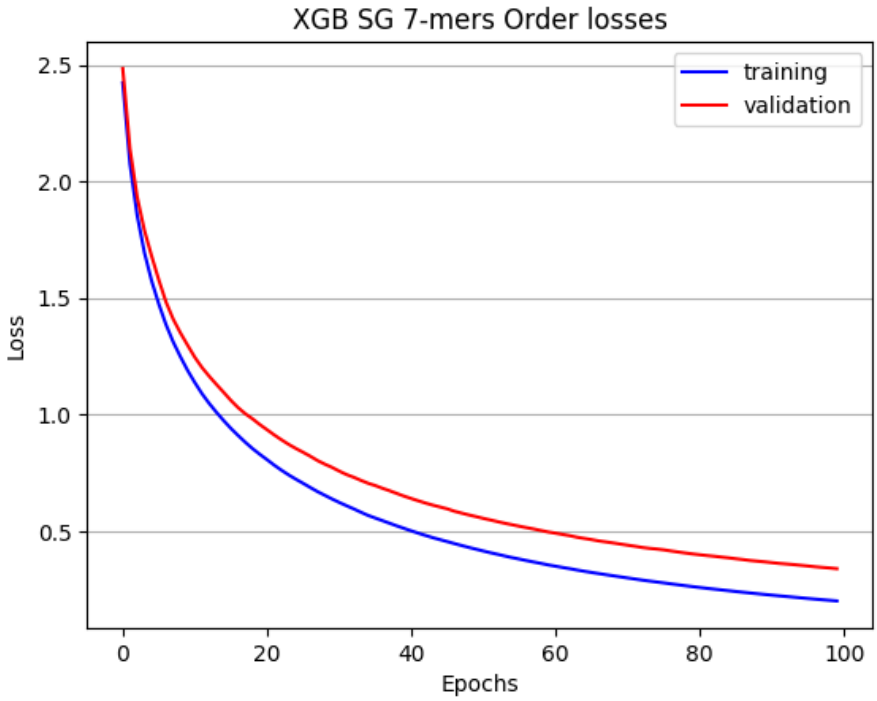

Figure 66: Training and validation loss curves for the XGBoost classifier tested with the SG dataset and 7-mers at order taxonomic rank.

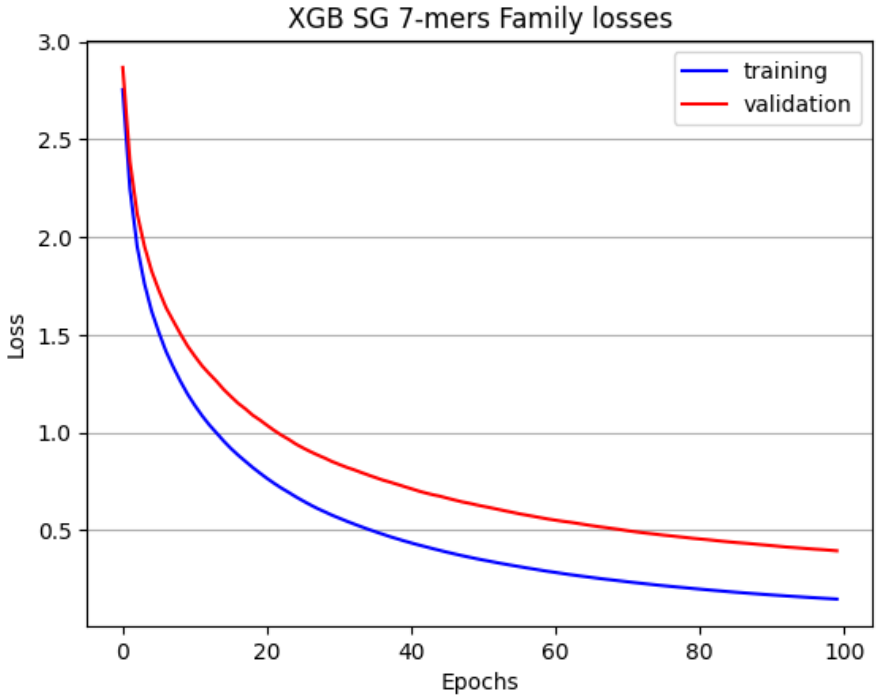

Figure 67: Training and validation loss curves for the XGBoost classifier tested with the SG dataset and 7-mers at family taxonomic rank.

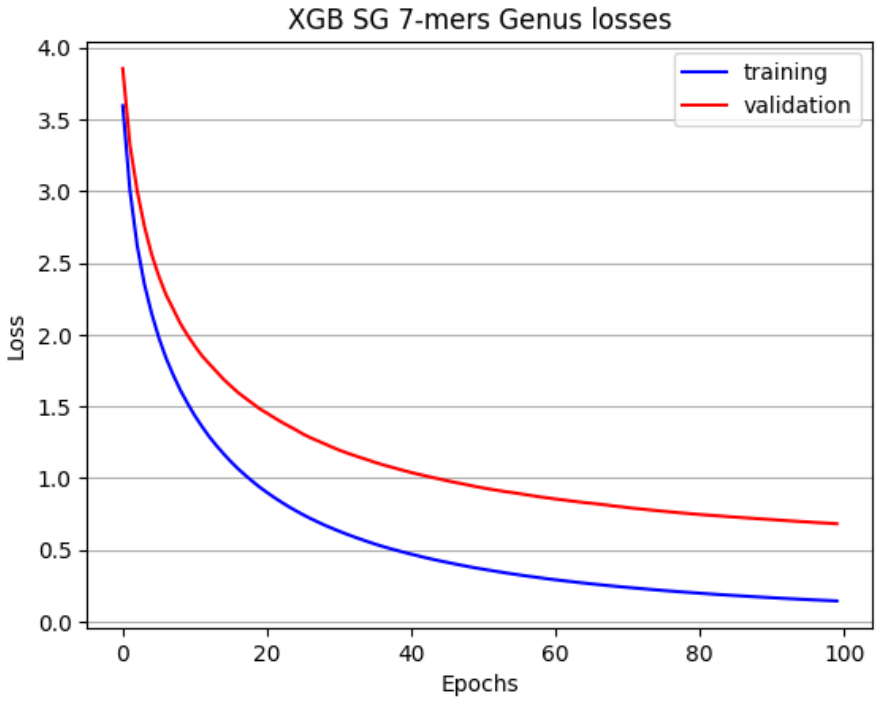

Figure 68: Training and validation loss curves for the XGBoost classifier tested with the SG dataset and 7-mers at genus taxonomic rank.

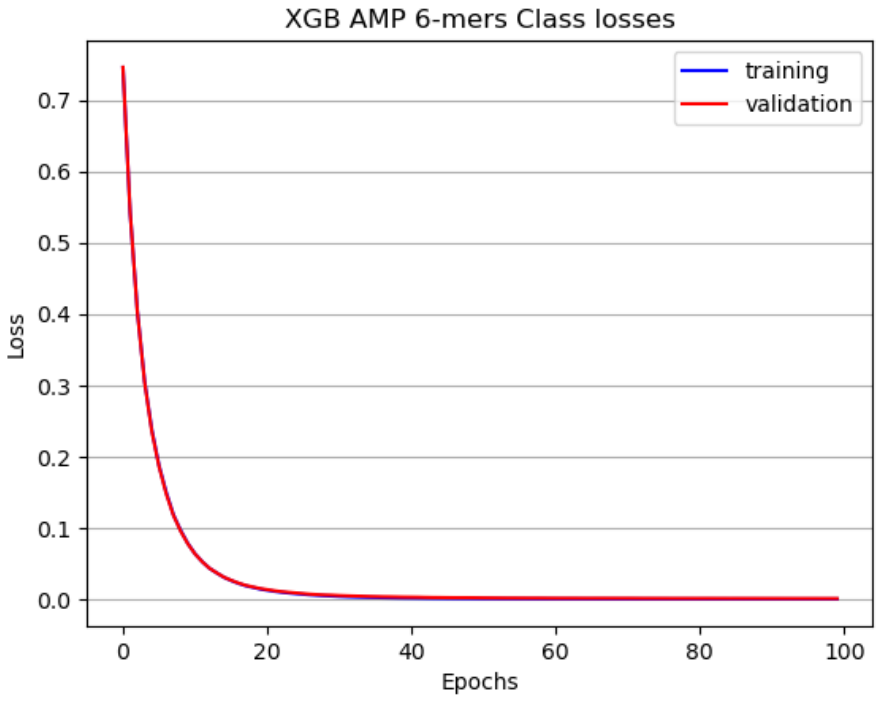

Figure 69: Training and validation loss curves for the XGBoost classifier tested with the AMP dataset and 6-mers at class taxonomic rank.

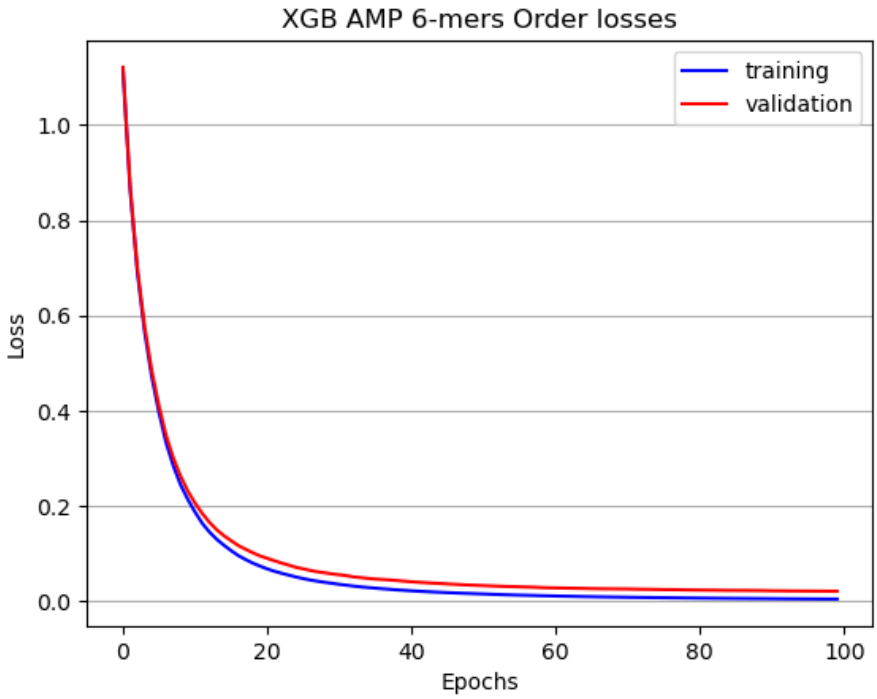

Figure 70: Training and validation loss curves for the XGBoost classifier tested with the AMP dataset and 6-mers at order taxonomic rank.

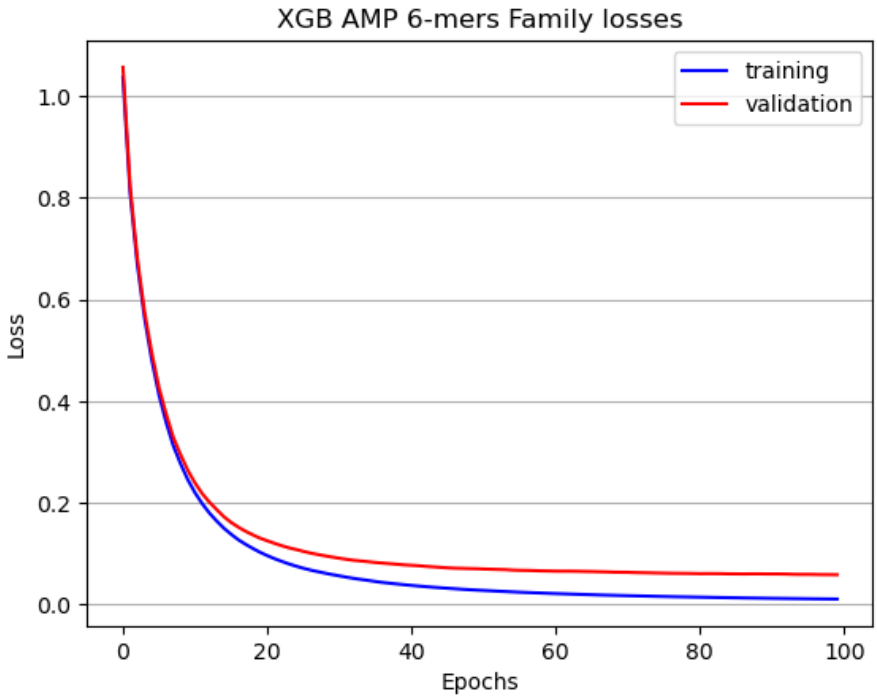

Figure 71: Training and validation loss curves for the XGBoost classifier tested with the AMP dataset and 6-mers at family taxonomic rank.

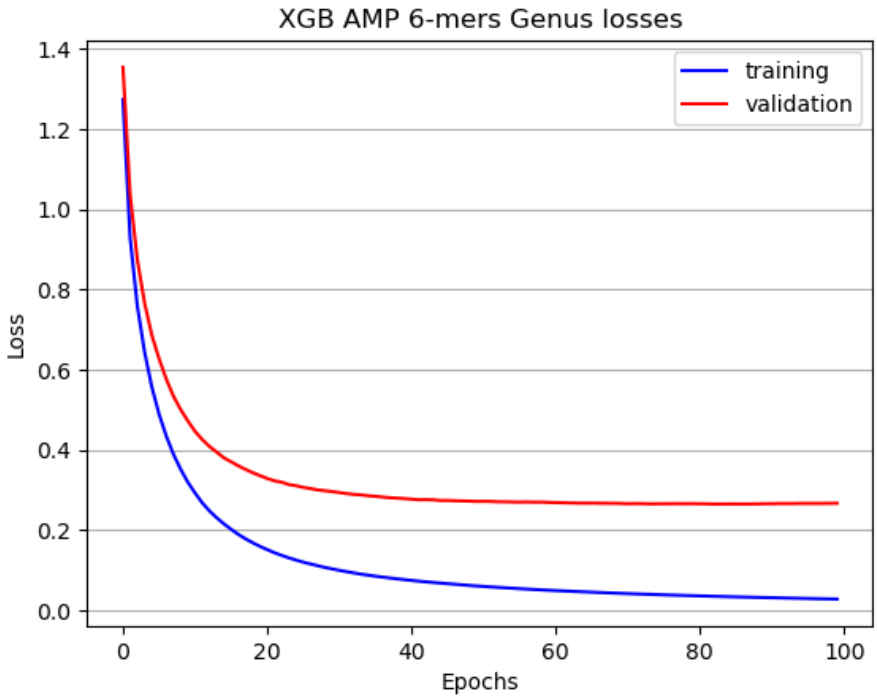

Figure 72: Training and validation loss curves for the XGBoost classifier tested with the AMP dataset and 6-mers at genus taxonomic rank.

3.3.6 Maximum tree depth hyperparameter

Regarding the parameters tuning, no significant effect has been observed changing the maximum tree depth from 3 to 2 and 4. As with the CNN model, the SG dataset with 7-mers and the AMP with 6-mers are the combinations selected for that experiment. Using the SG dataset, there is a slight performance decrease with a maximum tree depth of 2 as shown in Fig.73, however no change is appreciated using the AMP dataset as Fig.74 shows. All in all, the maximum tree depth does not seem to have a critical effect, although it may be slightly helpful to achieve a better classifier when using the SG dataset.

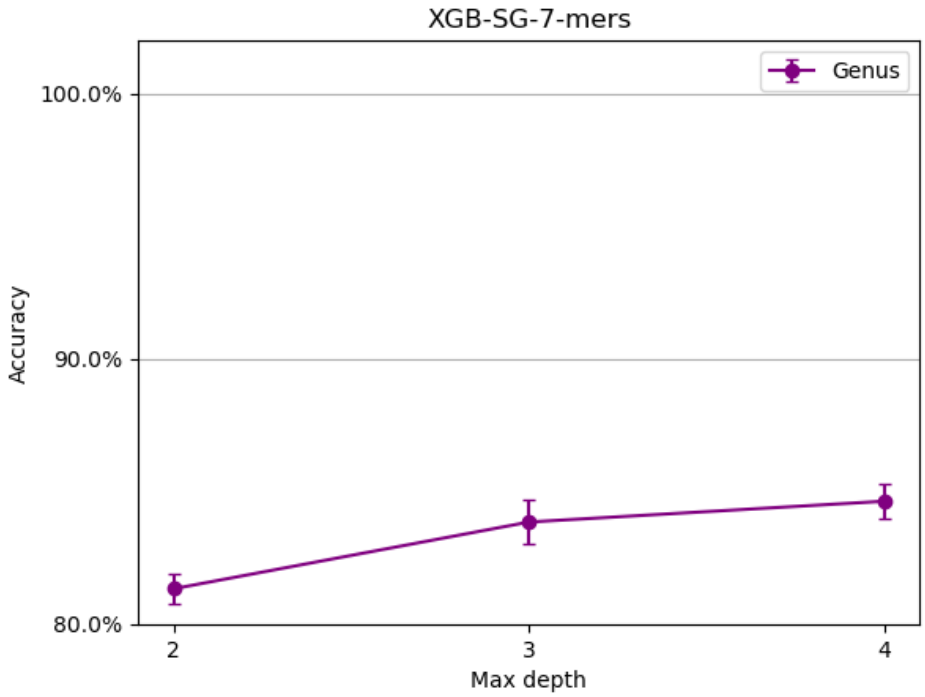

Figure 73: Accuracy of the XGBoost model with SG dataset with different maximum tree depth at genus level.

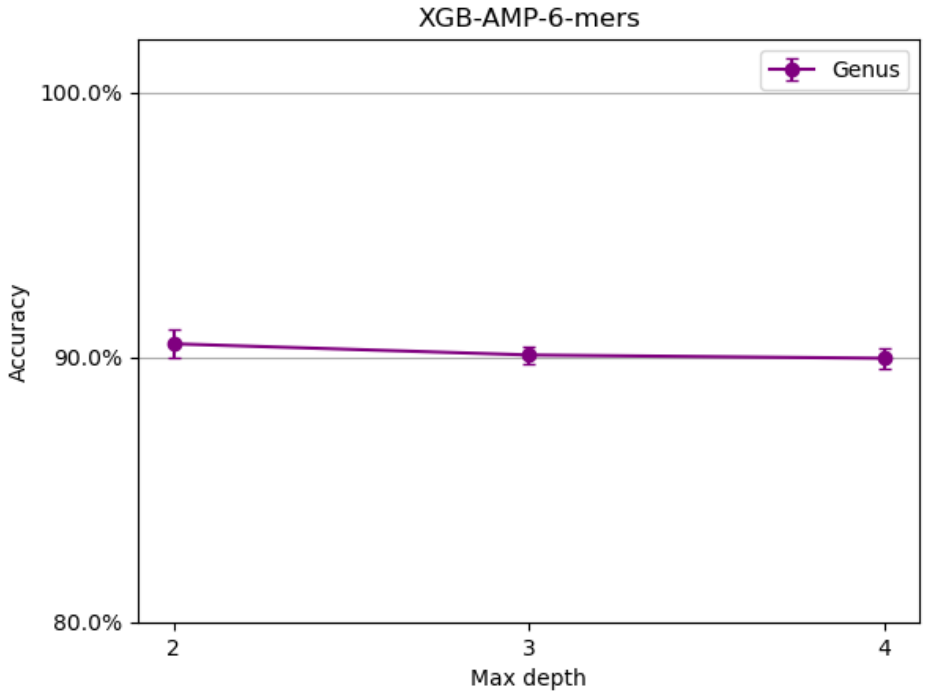

Figure 74: Accuracy of the XGBoost model with AMP dataset with different maximum tree depth at genus level.

3.4 Model comparison

Comparing the performance of the models on both synthetic datasets is indispensable to discern which is the better classifier in various situations. For this reason, all three classifiers, CNN, DBN and XGBoost, trained with the synthetic datasets are compared. However, since all methods clearly perform better with high k-mer size, only 6-mers and 7-mers based models are analysed. In Table.1 the accuracy of each created classifier for each dataset is shown. As it can be seen, at genus level the greatest accuracy is 91.5% achieved by the DBN model trained with 7-mers representation of the AMP dataset followed by the 91.4% of the CNN model with the same training dataset. In general, the models trained with the AMP dataset present higher accuracy than the ones trained with the SG dataset. Nevertheless, this may be due to further overfitting because the data are easier to distinguish. Furthermore, the nature of the SG and AMP dataset is not exactly the same, since AMP only use two hypervariable regions, V3 and V4, of the whole 16S rRNA sequences. For the SG dataset, the CNN classifier shows an 85.1% of accuracy at genus level, compared to the 81.0% of the DBN and the 83.8% of the XGBoost.

The CNN classifier seems slightly better than DBN and XGBoost regarding the accuracy achieved at genus level for both synthetic datasets. Also, it is competitive regardless of the taxonomic rank. The DBN classifier is strongly competitive when it is trained with the AMP but with the SG dataset at genus level is clearly worse than both, CNN and XGBoost. Then, the XGBoost is quite consistent in terms of the k-mer size since there is not an appreciable change of accuracy comparing 6-mers and 7-mers strategy, while the CNN and DBN classifiers experiment an improvement with the 7-mers strategy compared with 6-mers, specially when training with the SG dataset.

Train dataset	Taxonomic level	CNN (%)	DBN (%)	XGB (%)
SG 6-mers	Class	99.3±0.2	**99.6±0.1**	99.3±0.1
	Order	94.7±0.5	**95.7±0.3**	94.6±0.5
	Family	93.0±0.6	93.2±0.4	**93.8±0.4**
	Genus	80.7±1.2	77.5±0.8	**83.8±0.7**
SG 7-mers	Class	99.5±0.1	**99.7±0.1**	99.4±0.1
	Order	**96.8±0.4**	96.4±0.3	95.9±0.3
	Family	**95.7±0.6**	94.4±0.5	94.7±0.4
	Genus	**85.1±0.5**	81.0±0.5	83.8±0.9
AMP 6-mers	Class	**99.9±0.1**	**99.9±0.1**	**99.9±0.1**
	Order	**99.5±0.1**	**99.5±0.1**	99.4±0.1
	Family	97.4±0.4	97.4±0.3	**97.6±0.2**
	Genus	**90.8±0.3**	90.5±0.5	90.1±0.3
AMP 7-mers	Class	**99.9±0.1**	**99.9±0.1**	**99.9±0.1**
	Order	99.5±0.2	**99.7±0.2**	99.4±0.2
	Family	97.7±0.3	**97.8±0.2**	97.6±0.2
	Genus	91.4±0.5	**91.5±0.4**	90.2±0.4

Table 1: Accuracy comparison of the CNN, DBN and XGBoost models for the synthetic SG and AMP dataset and 6-mers and 7-mers representations. All taxonomic levels (class, order, family and genus) appears.

3.5 Testing

The testing step is an essential part of this work in order to evaluate public genomic databases with the created classifiers. Nonetheless, a critical part is to filter through existent public databases to form a reliable dataset of 16S rRNA sequences of known bacteria by the trained models. The selected models to be tested are the ones with the best accuracy. CNN-based model seems slightly better than both DBN and XGBoost taking into account the validation performed. For the SG dataset, the 7-mers strategy is clearly the best, while for the AMP dataset, the 6-mers strategy seems good enough compared to 7-mers. Nonetheless, finding representative 16S rRNA

compatible with the AMP dataset has not been possible since only hypervariable regions V3 and V4 of the 16S rRNA sequences are taken into account in such dataset. Therefore, the models trained with SG dataset and 7-mers representation are the ones used to test.

Regarding the testing datasets, TargetedLoci and Nucleotide16S, extracted or created using NCBI are suitable to be used. Moreover, the SILVA dataset is also adequate, as well as the FDA-ARGOS dataset. After filtering the datasets, 1330 sequences are kept from the TargetedLoci, 1475 from the Nucleotide16S, 13729 from SILVA and just 225 from FDA-ARGOS. In the first three datasets, 97 of the 100 genus levels known by the models are represented, while in FDA-ARGOS only 8 of them. The filtering is essential to discard sequences of species that are not included in the trained models. The success rate of predicting these public genomic datasets overcomes the 90% of accuracy, as data in Table.2 indicates. Other metrics, precision, recall and F1 score are also calculated and posted in Table.3. Specifically, using the CNN model, 94.89% of the sequences in the TargetedLoci dataset are well-predicted, while for the Nucleotide16S dataset the percentage is 91.46% and 93.40% for the SILVA dataset. Meanwhile, the DBN classifier, achieve a 93.01% of accuracy predicting data of the TargetedLoci dataset, 89.60% for the Nucleotide16S and 92.70% for SILVA dataset. Finally, the accuracies achieved by the XGBoost classifier are 94.73%, 91.12% and 91.35% for the TargetedLoci, Nucleotide16S and SILVA datasets respectively. Therefore, it can be observed that the DBN and XGBoost classifier have less accuracy, precision, recall and F1 score for most of the tested datasets than the CNN model. Furthermore, interestingly, all the filtered data in FDA-ARGOS are 100% well classified regardless of the model, however the number of samples is very little, and it can be misleading.

Classifier	**Train dataset**	**Test dataset**	**Correct**	**Total**	**Accuracy (%)**
CNN	SG 7-mers	TargetedLoci	1262	1330	**94.89**
	SG 7-mers	Nucleotide16S	1349	1475	**91.46**
	SG 7-mers	SILVA	12824	13729	**93.41**
	SG 7-mers	FDA-ARGOS	225	225	**100**
DBN	SG 7-mers	TargetedLoci	1237	1330	**93.01**
	SG 7-mers	Nucleotide16S	1323	1475	**89.69**
	SG 7-mers	SILVA	12541	13729	**92.70**
	SG 7-mers	FDA-ARGOS	225	225	**100**
XGBoost	SG 7-mers	TargetedLoci	1260	1330	**94.74**
	SG 7-mers	Nucleotide16S	1344	1475	**91.12**
	SG 7-mers	SILVA	12541	13729	**91.35**
	SG 7-mers	FDA-ARGOS	225	225	**100**

Table 2: Success rate (accuracy) of testing the CNN, DBN and XGBoost models, trained with the SG dataset and 7-mers strategy, predicting public genomic datasets named TargetedLoci, Nucleotide16S, SILVA and FDA-ARGOS.

Classifier	Train dataset	Test dataset	Precision (%)	Recall (%)	F1 score (%)
CNN	SG 7-mers	TargetedLoci	95.82	94.89	94.37
	SG 7-mers	Nucleotide16S	93.34	91.46	90.32
	SG 7-mers	SILVA	93.32	93.41	91.91
	SG 7-mers	FDA-ARGOS	100	100	100
DBN	SG 7-mers	TargetedLoci	94.74	93.01	92.15
	SG 7-mers	Nucleotide16S	92.35	89.69	88.61
	SG 7-mers	SILVA	93.60	92.70	91.22
	SG 7-mers	FDA-ARGOS	100	100	100
XGBoost	SG 7-mers	TargetedLoci	95.89	94.74	94.52
	SG 7-mers	Nucleotide16S	92.84	91.12	90.52
	SG 7-mers	SILVA	91.76	91.35	90.33
	SG 7-mers	FDA-ARGOS	100	100	100

Table 3: Precision, recall and F1 score of testing the CNN, DBN and XGBoost models, trained with the SG dataset and 7-mers strategy, predicting public genomic datasets named TargetedLoci, Nucleotide16S, SILVA and FDA-ARGOS.

Overall, the testing step for all three datasets is satisfactory. Nonetheless, knowing the whole accuracy gives not specific information about bacteria. The models may be quite efficient detecting some bacteria, at genus level, such as *Legionella*, but terrible in distinguishing other bacteria. For this reason, it may be interesting to list the bacteria genus which are worse detected, leading to a higher number of false negatives. Using the CNN classifier, the most interesting bacterial genus are shown in Table.4, Table.5 and Table.6 for the TargetedLoci, Nucleotide16S and SILVA datasets respectively with the correspondent number of true positives, false negatives and false positives. In Table.7, Table.8 and Table.9 there are the worst classified bacteria by the DBN algorithm. Equivalently, Table.10, Table.11 and Table.12 show the interesting tested bacterial genus with the XGBoost classifier.

TargetedLoci tested with CNN classifier			
Bacteria genus	**True Positives**	**False Negatives**	**False Positives**
Aquaspirillum	5	3	0
Falsochrobactrum	0	2	0
Halovibrio	1	2	0
Leisingera	5	3	1
Phaeobacter	6	3	0
Rhizobium	96	16	2
Sphingomonas	136	30	0

Table 4: Worse identified bacteria genus in the TargetedLoci when testing with the CNN classifier.

Nucleotide16S tested with CNN classifier			
Bacteria genus	**True Positives**	**False Negatives**	**False Positives**
Alteromonas	25	8	0
Aquaspirillum	5	4	0
Caulobacter	22	7	0
Falsochrobactrum	0	2	0
Halovibrio	1	2	0
Leisingera	5	3	1
Phaeobacter	6	3	0
Rhizobium	102	16	3
Sphingomonas	141	44	1
Stenotrophomonas	28	17	0

Table 5: Worse identified bacteria genus in the Nucleotide16S when testing with the CNN classifier.

SILVA tested with CNN classifier			
Bacteria genus	**True Positives**	**False Negatives**	**False Positives**
Falsochrobactrum	3	20	10
Leisingera	16	13	20
Loktanella	2	5	3
Nautella	22	6	8
Neisseria	1250	145	1
Phaeobacter	140	24	6
Pseudophaeobacter	22	15	3
Roseivivax	51	64	8
Sagittula	11	8	0
Sphingomonas	1345	465	1
Starkeya	10	4	0

Table 6: Worse identified bacteria genus in the SILVA when testing with the CNN classifier.

TargetedLoci tested with DBN classifier			
Bacteria genus	**True Positives**	**False Negatives**	**False Positives**
Aquaspirillum	5	3	0
Falsochrobactrum	0	2	0
Halovibrio	1	2	0
Leisingera	4	4	1
Phaeobacter	6	3	1
Rhizobium	77	35	0
Sphingomonas	138	28	0
Zymobacter	1	2	0

Table 7: Worse identified bacteria genus in the TargetedLoci when testing with the DBN classifier.

Nucleotide16S tested with DBN classifier			
Bacteria genus	**True Positives**	**False Negatives**	**False Positives**
Alteromonas	25	8	0
Aquaspirillum	5	4	0
Caulobacter	22	7	0
Falsochrobactrum	0	2	0
Halovibrio	1	2	0
Leisingera	4	4	1
Marinomonas	37	7	0
Phaeobacter	6	3	1
Rhizobium	83	35	0
Sphingomonas	141	44	1
Stenotrophomonas	28	17	0

Table 8: Worse identified bacteria genus in the Nucleotide16S when testing with the DBN classifier.

SILVA tested with DBN classifier			
Bacteria genus	**True Positives**	**False Negatives**	**False Positives**
Coxiella	323	63	0
Falsochrobactrum	0	23	1
Labrys	41	12	0
Leisingera	11	18	24
Loktanella	1	6	1
Nautella	22	6	29
Pelagibius	8	36	0
Phaeobacter	141	23	6
Pseudophaeobacter	14	23	1
Roseivivax	80	35	8
Sagittula	10	9	0
Sphingomonas	1284	526	1

Table 9: Worse identified bacteria genus in the SILVA when testing with the DBN classifier.

TargetedLoci tested with XGBoost classifier			
Bacteria genus	**True Positives**	**False Negatives**	**False Positives**
Ancylobacter	7	4	0
Aquaspirillum	5	3	0
Falsochrobactrum	0	2	0
Halovibrio	1	2	0
Leisingera	1	7	3
Phaeobacter	4	5	0
Rhizobium	94	18	2
Rhizorhabdus	6	2	6
Rhodopseudomonas	13	3	0

Table 10: Worse identified bacteria genus in the TargetedLoci when testing with the XGBoost classifier.

Nucleotide16S tested with XGBoost classifier			
Bacteria genus	**True Positives**	**False Negatives**	**False Positives**
Alteromonas	25	8	0
Acylobacter	7	4	0
Aquaspirillum	5	4	0
Caulobacter	22	7	0
Falsochrobactrum	0	2	0
Halovibrio	1	2	0
Leisingera	1	7	3
Phaeobacter	4	5	0
Rhizobium	100	18	2
Rhodopseudomonas	17	8	1
Sphingomonas	155	30	3
Stenotrophomonas	28	17	0

Table 11: Worse identified bacteria genus in the Nucleotide16S when testing with the XGBoost classifier.

SILVA tested with XGBoost classifier			
Bacteria genus	**True Positives**	**False Negatives**	**False Positives**
Falsochrobactrum	1	22	1
Leisingera	4	25	106
Loktanella	2	5	16
Nautella	22	6	12
Neisseria	1248	147	7
Phaeobacter	52	112	6
Ponticoccus	18	6	11
Pseudophaeobacter	19	18	6
Rhodopseudomonas	72	50	3
Roseivivax	30	85	15
Sagittula	9	10	1
Sphingomonas	1523	287	2
Starkeya	9	5	6

Table 12: Worse identified bacteria genus in the SILVA when testing with the XGBoost classifier.

Analysing bacteria, for which the models have more problems to predict correctly, it can be seen that are quite similar regardless of the classifier used. However, in general, the CNN seems to perform better than the DBN and XGBoost classifiers for the three testing datasets taking into account the different metrics. Interestingly, on one hand, some bacteria genus such as *Phaeobacter* or *Leisingera* appears in the confusion matrix plotted when validating the models (for example Fig.20). Thereby, the errors regarding the testing may be explained by poor learning of the classifiers. On the other hand, for instance *Sphingomonas* or *Neisseria* present a non-negligible number of false negatives, although the training and validating steps seem adequate. What is more, all *Sphingomonas* and *Neisseria* in the curated genomic FDA-ARGOS dataset are well detected. Therefore, it is possible that some sequences of these bacteria are mislabelled in the NCBI and SILVA public genomic databases. Finally,

it is hard to determine the origin of the misclassifications for all the other bacterial genus, although the most likely cause is classifier errors due to the nature of the input data.

Chapter 4

Discussion

Different datasets of 16S ribosomal RNA gene sequences of bacteria and machine learning (or deep learning) algorithms have been studied to achieve the best classifier to distinguish bacteria at genus taxonomic rank. First, the synthetic datasets, named SG and AMP, have been used to train CNN, DBN and XGBoost models. Then, the models trained with SG dataset and 7-mers have been used to test and evaluate the datasets extracted from NCBI, SILVA and FDA-ARGOS databases. The strategy followed in all cases of this master thesis is based on the k-mer frequency. Actually, the selection of the k-mer size has been proven critical according to both, accuracy (and other metrics) of the model and computing time. The classifiers' accuracy with small k-mer size is generally poor, while for 6-mer and 7-mer the accuracy remains almost constant, especially for the AMP dataset. CNN, DBN and XGBoost models have been trained with k-mer size of 3 to 7.

In terms of time, the training time exponentially increases, making it almost unfeasible to explore larger k-mer size. Furthermore, logically, this exponential rise is significantly notable comparing k-mer size of 6 and 7. Therefore, taking into account the accuracy and computing time, 6-mers seems a satisfactory option, although with 7-mer the accuracy is slightly higher for the CNN, DBN and XGBoost models. Interestingly, XGBoost classifier is the fastest to complete the training process for a k-mer size lower than 6, while for 6-mers and 7-mers CNN is the quickest. It is also

noticeable that the taxonomic rank has an impact concerning the computing time of the XGBoost algorithm, while it has no notable effect in the deep learning algorithms. Finally, it may be highlighted that the DBN classifier is clearly the slowest, and it does not achieve better accuracy scores than the other models.

Regarding the classifier accuracies, all three classifiers work successfully at the taxonomic level of class, order and family and present more difficulties at genus level, although the accuracy achieved is still high. In general, the XGBoost model does not have much to envy to the deep learning approach classifiers. In fact, the XGBoost is less sensible to the dataset used (SG or AMP), while, the CNN and DBN are slightly better when using the AMP dataset. In terms of accuracy, the CNN algorithm narrowly outperforms both DBN and XGBoost, also in regard to the training time is a competitive algorithm. Therefore, the CNN classifier seems better than the DBN and XGBoost. Moreover, in terms of precision, recall, F1-score and AUC, the CNN model is also slightly better than the DBN and XGBoost.

Comparing the dataset performance, the AMP seems slightly better than SG. The AMP achieves higher accuracy and takes less time, however it may have more overfitting since it is easier to understand and distinguish the data with few epochs as the loss curves have shown. Nevertheless, it should also be noted that the testing part is, actually, extremely important and the SG-trained models perform quite successfully for all the selected testing datasets. It has to be noted that the TargetedLoci, Nucleotide16S, SILVA and FDA-ARGOS datasets have complete 16S rRNA sequences and therefore are compatible for models trained with the SG dataset but not with the AMP. Specifically, the testing datasets to evaluate NCBI and SILVA present more than a 90% accuracy while FDA-ARGOS has a 100% of accuracy, but with significant fewer data. Most of the misclassification can be explained by classifiers' errors. However, the case of *Sphingomonas* is especially remarkable since more misclassification than expected are found regarding NCBI and SILVA databases. Also, *Neisseria* genus rank in SILVA database has a significant number of misclassifications. Furthermore, both *Sphingomonas* and *Neisseria* appears in FDA-ARGOS and they have no misclassification.

Finally, it is essential to mention the importance of achieving high precise models. For instance, assuming that an algorithm has 99.0% of accuracy achieved with a learning dataset of approximately 28000 sample, and a validation dataset of 2800 samples. Then, if that algorithm is trained again with a larger dataset and tested for 1 million samples, the number of misclassifications would be around 10000 considering that the accuracy is maintained. Obviously, this may be dangerous, thereby a model must perform excellently to minimise the number of misclassifications and to be considered for real world applications.

4.1 Limitations

- **Number of bacterial genus:** The number of bacterial genus in the SG synthetic dataset is 100. However, not all the genus appear in the NCBI, SILVA and FDA-ARGOS. In contrast, NCBI, SILVA and FDA-ARGOS have many sequences representing other bacterial genus which can not be used since the models trained with the SG synthetic dataset do not know them. Not having the same genus in each dataset implies having to filter them for compatibility, losing a not inconsiderable amount of data.

- **16S rRNA:** Using only the 16S rRNA gene to classify bacteria can not be enough to distinguish correctly between bacteria at genus level. The 16S rRNA gene is well conserved, however it is similar between some different bacteria species, thereby it is relatively easy to have misclassifications.

- **Computing time:** One of the main important impediments to train a classifier has been the computing time. Predictably, the larger the k-mer size, the longer the process takes. Especially, for the DBN model, this problem is notable.

- **Large amount of samples:** Another key point is the quantity of sequences needed to train the algorithm. Theoretically, the greater amount of samples for each bacterium, the best the model would perform as long as there is no critical overfitting. The presence of many samples may lead to memory issues

and the computing time issue already mentioned. Therefore, a balanced have to be reached.

4.2 Future work

- The datasets used are not representative of all bacteria. Then, future work could generate a more complete dataset in which more genus are represented. Also, more samples of each bacterium can help the classifier to distinguish the new unknown sequences. For instance, Hoarfrost et al. [38] use an appropriate complete dataset of marine metagenomes.

- Using a set of genes instead of only the 16S rRNA would help distinguish bacteria at genus level. A new possible approach is to concatenate the sequences of various genes [39]. This approach would lead to larger synthetic sequences with the advantage of using the information of more than one gene.

- Creating a large dataset taking into account the AMP dataset and therefore only V3-V4 hypervariable regions. Hence, enough data to train, validate and test models would be available.

- The k-mer strategy has been proven quite efficient, however it may not be sufficient to achieve even better accuracies. Each k-mer represents a feature, however, using their frequency, the information of the position of each nucleotide in the sequence is lost. For this reason, perhaps, a strategy considering both, the location and frequency of the k-mer can be used. Another option may be to consider consecutive k-mer sizes (1-mer, 2-mer, etc) but then the complexity of the task significantly increases.

- The model used as classifier may be improved. Deep learning approach offers an incredible amount of options. For instance, exploring the possibility of using Bidirectional Encoder Representations from Transformers (BERT) also known simply as transformers can be interesting to improve the task success.

4.3 Conclusions

- AMP dataset seems slightly better than SG dataset regarding the accuracy in the validation step. A higher accuracy is achieved in all classifiers, especially at the genus level, but due to a further overfitting. Furthermore, it requires less time to train a model. Nonetheless, finding amplicon 16S rRNA sequences keeping only V3-V4 hypervariable regions is hard. Additionally, the testing of the SG-trained models with 7-mers representation with public genomic databases was successful. For this reason, both SG and AMP datasets may be useful, since none of them clearly outperforms the other.

- The CNN classifier is slightly better than the DBN and XGBoost when using the SG and AMP dataset. In terms of computing time is also competitive enough. Furthermore, regarding the testing with data in public genomic databases, the CNN-based model is also the one with the best accuracy.

- The k-mer size has been proven critical, the strategy of 6-mers and 7-mers are the best. By using a higher k-mer size than 7, the performance metrics would not rise significantly, whereas the computing time would greatly increase.

- The process of tuning hyperparameters have not shown any significant performance change for neither the CNN nor the XGBoost model. The impact made appears to be nearly unnoticed.

- Public genomic databases are in general reliable. Most of the misclassifications can be understood considering the classifiers' errors. However, NCBI and SILVA database may have some errors in specific bacteria genus level such as *Sphingomonas* and *Neisseria* taking into account that the models have learned correctly to classify these bacteria and that FDA-ARGOS has no misclassification errors.

List of Figures

List of Tables

Bibliography

[1] Culligan, E. P., Sleator, R. D., Marchesi, J. R. & Hill, C. Metagenomics and novel gene discovery: Promise and potential for novel therapeutics. *Virulence* **5** (2014).

[2] Wooley, J. C. & Ye, Y. Metagenomics: Facts and artifacts, and computational challenges. *JOURNAL OF COMPUTER SCI-ENCE AND TECHNOLOGY* **25**, 71–81 (2010).

[3] Simon, C. & Daniel, R. Metagenomic analyses: Past and future trends. *Applied and Environmental Microbiology* **77**, 1153–1161 (2011).

[4] Ereshefsky, M. *The Poverty of the Linnaean Hierarchy: A Philosophical Study of Biological Taxonomy* (Cambridge University Press, 2000).

[5] Mardis, E. R. Anticipating the $1,000 genome. *Genome Biology* **7** (2006).

[6] Tringe, S. G. & Hugenholtz, P. A renaissance for the pioneering 16s rrna gene. *Current Opinion in Microbiology* **11**, 442–446 (2008).

[7] Wang, Y. & Qian, P.-Y. Conservative fragments in bacterial 16s rrna genes and primer design for 16s ribosomal dna amplicons in metagenomic studies. *PLoS ONE* **4**, e7401 (2009). URL `https://dx.plos.org/10.1371/journal.pone.0007401`.

[8] Fiannaca, A. *et al.* Deep learning models for bacteria taxonomic classification of metagenomic data. *BMC Bioinformatics* **19** (2018).

[9] Wang, Q., Garrity, G. M., Tiedje, J. M. & Cole, J. R. Naïve bayesian classifier for rapid assignment of rrna sequences into the new bacterial taxonomy. *Applied and Environmental Microbiology* **73**, 5261–5267 (2007).

[10] Chen, T. & Guestrin, C. Xgboost: A scalable tree boosting system. *Proceedings of the ACM SIGKDD International Conference on Knowledge Discovery and Data Mining* **13-17-August-2016**, 785–794 (2016).

[11] Piano, S. L. Ethical principles in machine learning and artificial intelligence: cases from the field and possible ways forward. *Humanities and Social Sciences Communications* **7** (2020).

[12] Sarker, I. H. Machine learning: Algorithms, real-world applications and research directions. *SN Computer Science* **2** (2021).

[13] Community cleverness required. *Nature* **455**, 1 (2008). URL `https://doi.org/10.1038/455001a`.

[14] Xu, C. & Jackson, S. A. Machine learning and complex biological data. *Genome Biology* **20** (2019).

[15] Stephens, Z. D. *et al.* Big data: Astronomical or genomical? *PLoS Biology* **13** (2015).

[16] Pallen, M. J. Diagnostic metagenomics: Potential applications to bacterial, viral and parasitic infections. *Parasitology* **141**, 1856–1862 (2014).

[17] Amrane, S. *et al.* Metagenomic and culturomic analysis of gut microbiota dysbiosis during clostridium difficile infection. *Scientific Reports* **9** (2019).

[18] Mardanov, A. V., Kadnikov, V. V. & Ravin, N. V. Metagenomics: A paradigm shift in microbiology. *Metagenomics: Perspectives, Methods, and Applications* 1–13 (2018).

[19] Pérez-Cobas, A. E., Gomez-Valero, L. & Buchrieser, C. Metagenomic approaches in microbial ecology: an update on whole-genome and marker gene sequencing analyses. *Microbial Genomics* **6** (2020).

[20] Bharagava, R. N., Purchase, D., Saxena, G. & Mulla, S. I. Applications of metagenomics in microbial bioremediation of pollutants: From genomics to environmental cleanup. *Microbial Diversity in the Genomic Era* 459–477 (2019).

[21] Martínez-Porchas, M. & Vargas-Albores, F. Microbial metagenomics in aquaculture: a potential tool for a deeper insight into the activity. *Reviews in Aquaculture* **9**, 42–56 (2017). URL `https://onlinelibrary.wiley.com/doi/abs/10.1111/raq.12102`. `https://onlinelibrary.wiley.com/doi/pdf/10.1111/raq.12102`.

[22] Pereira, F. Metagenomics: A gateway to drug discovery. *Advances in Biological Science Research: A Practical Approach* 453–468 (2019).

[23] Burman, W. J. & Reves, R. R. Review of False-Positive Cultures for Mycobacterium tuberculosis and Recommendations for Avoiding Unnecessary Treatment. *Clinical Infectious Diseases* **31**, 1390–1395 (2000). URL `https://doi.org/10.1086/317504`. `https://academic.oup.com/cid/article-pdf/31/6/1390/1048663/31-6-1390.pdf`.

[24] Fischer, G. W., Longfield, R., Hemming, V. G., Valdes-Dapena, A. & Patrick Smith, L. Pneumococcal Sepsis with False-Negative Blood Cultures. *American Journal of Clinical Pathology* **78**, 348–350 (1982). URL `https://doi.org/10.1093/ajcp/78.3.348`. `https://academic.oup.com/ajcp/article-pdf/78/3/348/26429550/ajcpath78-0348.pdf`.

[25] Afshinnekoo, E. *et al.* Geospatial resolution of human and bacterial diversity with city-scale metagenomics. *Cell Systems* **1**, 72–87 (2015).

[26] Acland, A. *et al.* Database resources of the national center for biotechnology information. *Nucleic Acids Research* **41** (2013).

[27] Quast, C. *et al.* The silva ribosomal rna gene database project: Improved data processing and web-based tools. *Nucleic Acids Research* **41** (2013).

[28] Sichtig, H. *et al.* Fda-argos is a database with public quality-controlled reference genomes for diagnostic use and regulatory science. *Nature Communications* **10** (2019).

[29] Kunin, V., Copeland, A., Lapidus, A., Mavromatis, K. & Hugenholtz, P. A bioinformatician's guide to metagenomics. *Microbiology and Molecular Biology Reviews* **72**, 557–578 (2008).

[30] Tonkovic, P. *et al.* Literature on applied machine learning in metagenomic classification: A scoping review. *Biology* **9**, 1–25 (2020).

[31] Liang, Q., Bible, P. W., Liu, Y., Zou, B. & Wei, L. Deepmicrobes: taxonomic classification for metagenomics with deep learning. *NAR Genomics and Bioinformatics* **2** (2020). URL `https://academic.oup.com/nargab/article/2/1/lqaa009/5740226`.

[32] Hsu, Y.-F. *et al.* *Deep Learning Approach for Pathogen Detection Through Shotgun Metagenomics Sequence Classification*, 24–30 (Springer International Publishing, Cham, 2019).

[33] Tjärnström, E. & Granholm, N. *Metagenomic Classification using Machine Learning Applied to SARS-CoV-2 and Viruses.* Master's thesis, Umeå University (2020).

[34] Yuan, C., Lei, J., Cole, J. & Sun, Y. Reconstructing 16s rrna genes in metagenomic data. In *Bioinformatics*, vol. 31, i35–i43 (Oxford University Press, 2015).

[35] Ramazzotti, M., Berná, L., Donati, C. & Cavalieri, D. riboframe: An improved method for microbial taxonomy profiling from non-targeted metagenomics. *Frontiers in genetics* **6**, 329 (2015). URL `www.frontiersin.org`.

[36] Krizhevsky, A., Sutskever, I. & Hinton, G. E. Imagenet classification with deep convolutional neural networks. In Pereira, F., Burges, C. J. C., Bottou, L. & Weinberger, K. Q. (eds.) *Advances in Neural Information Processing Systems*, vol. 25 (Curran Associates, Inc., 2012). URL `https://proceedings.neurips.cc/paper/2012/file/c399862d3b9d6b76c8436e924a68c45b-Paper.pdf`.

[37] Hinton, G. E., Osindero, S. & Teh, Y.-W. A Fast Learning Algorithm for Deep Belief Nets. *Neural Computation* **18**, 1527–1554 (2006). URL `https://doi.org/10.1162/neco.2006.18.7.1527`. `https://direct.mit.edu/neco/article-pdf/18/7/1527/816558/neco.2006.18.7.1527.pdf`.

[38] Hoarfrost, A., Aptekmann, A., Farfañuk, G. & Bromberg, Y. Deep learning of a bacterial and archaeal universal language of life enables transfer learning and illuminates microbial dark matter. *Nature Communications* **13**, 2606 (2022). URL `https://www.nature.com/articles/s41467-022-30070-8`.

[39] Hutchison, C. A. *et al.* Design and synthesis of a minimal bacterial genome. *Science* **351** (2016).

ChapterAppendix

Appendix A

Supplementary tables and figures

In this appendix the exact values of accuracy, training and inference computing time, precision, recall, F1 score and Area Under the Curve metrics are presented. Also, some extra confusion matrix are shown. First, the values for the CNN model appears, then for the DBN classifier and at the end for the XGBoost algorithm.

A.1 CNN supplementary results

k-mer size	**SG Accuracy**			
	Class	**Order**	**Family**	**Genus**
3	0.806±0.005	0.472±0.011	0.374±0.010	0.170±0.012
4	0.936±0.004	0.680±0.009	0.569±0.020	0.354±0.008
5	0.982±0.003	0.852±0.008	0.799±0.012	0.617±0.021
6	0.993±0.002	0.947±0.005	0.930±0.006	0.806±0.012
7	0.995±0.001	0.968±0.004	0.957±0.006	0.851±0.005

Table A.1: Accuracy of the CNN model with SG dataset.

k-mer size	AMP Accuracy			
	Class	**Order**	**Family**	**Genus**
3	0.937±0.006	0.765±0.013	0.671±0.010	0.507±0.012
4	0.991±0.001	0.945±0.004	0.889±0.008	0.774±0.015
5	0.998±0.001	0.986±0.002	0.959±0.001	0.882±0.007
6	0.999±0.001	0.995±0.001	0.974±0.004	0.908±0.003
7	0.999±0.001	0.995±0.002	0.977±0.003	0.914±0.005

Table A.2: Accuracy of the CNN model with AMP dataset.

k-mer size	SG			
	Training time (s)			
	Class	**Order**	**Family**	**Genus**
3	351±6	353±4	341±5	346±6
4	352±6	343±4	346±3	350±6
5	353±5	350±5	348±4	350±5
6	380±30	357±6	380±20	380±20
7	461±6	461±6	470±10	463±6

Table A.3: Training time in seconds (s) for the CNN model and SG dataset.

k-mer size	AMP			
	Training time (s)			
	Class	**Order**	**Family**	**Genus**
3	292±2	297±3	299±3	296±2
4	300±1	296±2	301±3	300±3
5	298±3	302±2	299±3	299±3
6	311±4	313±4	310±3	313±4
7	450±7	451±5	449±6	450±6

Table A.4: Training time in seconds (s) for the CNN model and AMP dataset.

k-mer size	**SG**			
	Inference time (s)			
	Class	**Order**	**Family**	**Genus**
3	0.6±0.8	0.6±0.8	0.6±0.8	0.6±0.8
4	0.7±1.0	0.6±0.8	0.6±0.7	0.6±0.8
5	0.5±0.6	0.6±0.7	0.5±0.7	0.6±0.7
6	0.6±0.7	0.7±0.8	0.6±0.7	0.6±0.6
7	0.8±1.3	0.8±1.1	0.9±1.5	0.7±1.1

Table A.5: Inference time in seconds (s) for the CNN model and SG dataset.

k-mer size	**AMP**			
	Inference time (s)			
	Class	**Order**	**Family**	**Genus**
3	0.23±0.05	0.24±0.06	0.24±0.05	0.22±0.06
4	0.24±0.05	0.24±0.05	0.25±0.05	0.24±0.04
5	0.24±0.04	0.25±0.04	0.24±0.06	0.23±0.06
6	0.3±0.1	0.29±0.07	0.30±0.05	0.29±0.06
7	1.2±1.5	1.0±1.4	1.1±1.5	1.3±1.6

Table A.6: Inference time in seconds (s) for the CNN model and AMP dataset.

Dataset	Precision CNN			
	Class	**Order**	**Family**	**Genus**
SG_k3	0.801±0.006	0.467±0.016	0.355±0.013	0.170±0.013
SG_k4	0.936±0.004	0.688±0.011	0.567±0.021	0.359±0.007
SG_k5	0.982±0.003	0.857±0.008	0.805±0.011	0.627±0.021
SG_k6	0.993±0.002	0.948±0.005	0.932±0.006	0.812±0.012
SG_k7	0.995±0.001	0.968±0.004	0.958±0.006	0.858±0.005
AMP_k3	0.937±0.006	0.768±0.012	0.668±0.010	0.512±0.013
AMP_k4	0.991±0.001	0.946±0.004	0.890±0.008	0.777±0.015
AMP_k5	0.998±0.001	0.986±0.002	0.959±0.001	0.884±0.007
AMP_k6	0.999±0.001	0.995±0.001	0.974±0.003	0.911±0.004
AMP_k7	0.999±0.001	0.995±0.002	0.977±0.003	0.918±0.007

Table A.7: Precision metric for the CNN model.

Dataset	Recall CNN			
	Class	**Order**	**Family**	**Genus**
SG_k3	0.806±0.005	0.472±0.011	0.374±0.010	0.170±0.012
SG_k4	0.936±0.004	0.680±0.009	0.569±0.020	0.354±0.008
SG_k5	0.982±0.003	0.852±0.008	0.799±0.012	0.617±0.021
SG_k6	0.993±0.002	0.947±0.005	0.930±0.006	0.806±0.012
SG_k7	0.995±0.001	0.968±0.004	0.957±0.006	0.851±0.005
AMP_k3	0.937±0.006	0.765±0.013	0.671±0.010	0.507±0.012
AMP_k4	0.991±0.001	0.945±0.004	0.889±0.008	0.774±0.015
AMP_k5	0.998±0.001	0.986±0.002	0.959±0.001	0.882±0.007
AMP_k6	0.999±0.001	0.995±0.001	0.974±0.004	0.908±0.003
AMP_k7	0.999±0.001	0.995±0.002	0.977±0.003	0.914±0.005

Table A.8: Recall metric for the CNN model.

Dataset	F1 Score CNN			
	Class	**Order**	**Family**	**Genus**
SG_k3	0.800±0.005	0.444±0.011	0.337±0.012	0.168±0.013
SG_k4	0.935±0.004	0.665±0.010	0.551±0.020	0.352±0.008
SG_k5	0.982±0.003	0.847±0.008	0.793±0.014	0.615±0.021
SG_k6	0.993±0.002	0.946±0.005	0.930±0.006	0.805±0.012
SG_k7	0.995±0.001	0.967±0.004	0.957±0.006	0.850±0.005
AMP_k3	0.937±0.006	0.759±0.014	0.665±0.010	0.504±0.012
AMP_k4	0.991±0.001	0.944±0.004	0.888±0.008	0.772±0.015
AMP_k5	0.998±0.001	0.986±0.002	0.958±0.001	0.881±0.007
AMP_k6	0.999±0.001	0.995±0.001	0.973±0.003	0.907±0.003
AMP_k7	0.999±0.001	0.995±0.002	0.977±0.004	0.913±0.006

Table A.9: F1 score metric for the CNN model.

Dataset	Area Under the Curve CNN			
	Class	**Order**	**Family**	**Genus**
SG_k3	0.904±0.004	0.866±0.009	0.865±0.005	0.862±0.008
SG_k4	0.988±0.002	0.955±0.004	0.947±0.008	0.945±0.003
SG_k5	0.998±0.001	0.991±0.001	0.991±0.001	0.986±0.001
SG_k6	0.999±0.001	0.999±0.001	0.999±0.001	0.996±0.001
SG_k7	0.999±0.001	0.999±0.001	0.999±0.001	0.997±0.001
AMP_k3	0.987±0.003	0.979±0.002	0.975±0.001	0.980±0.001
AMP_k4	0.999±0.001	0.998±0.001	0.997±0.001	0.997±0.001
AMP_k5	0.999±0.001	0.999±0.001	0.999±0.001	0.999±0.001
AMP_k6	0.999±0.001	0.999±0.001	0.999±0.001	0.999±0.001
AMP_k7	0.999±0.001	0.999±0.001	0.999±0.001	0.999±0.001

Table A.10: Area under the curve metric for the CNN model.

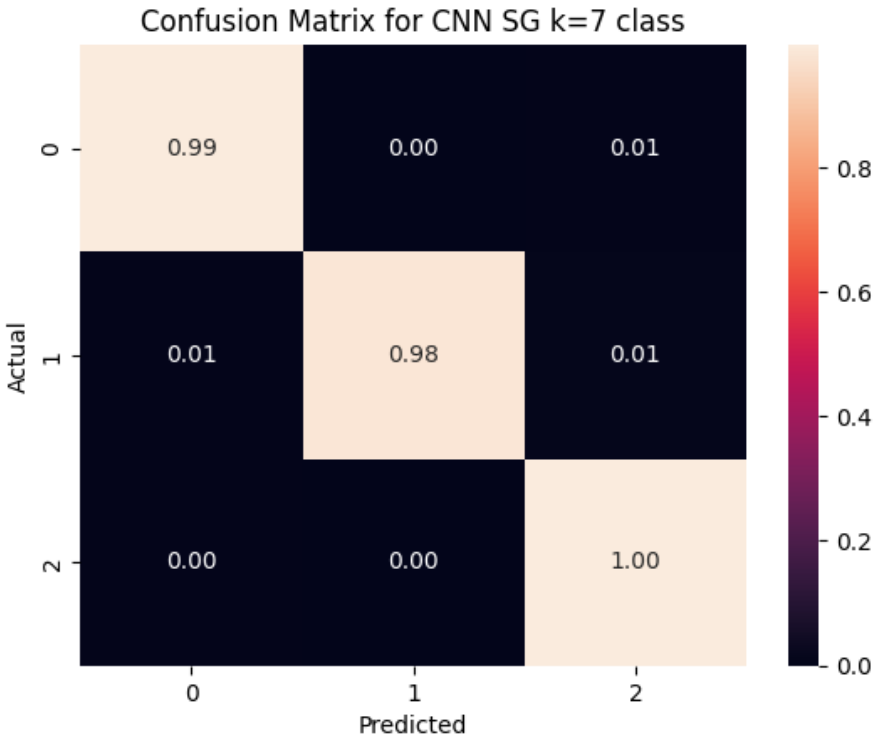

Figure A.1: Confusion matrix of the CNN model at the class level, trained with the SG dataset and 7-mers.

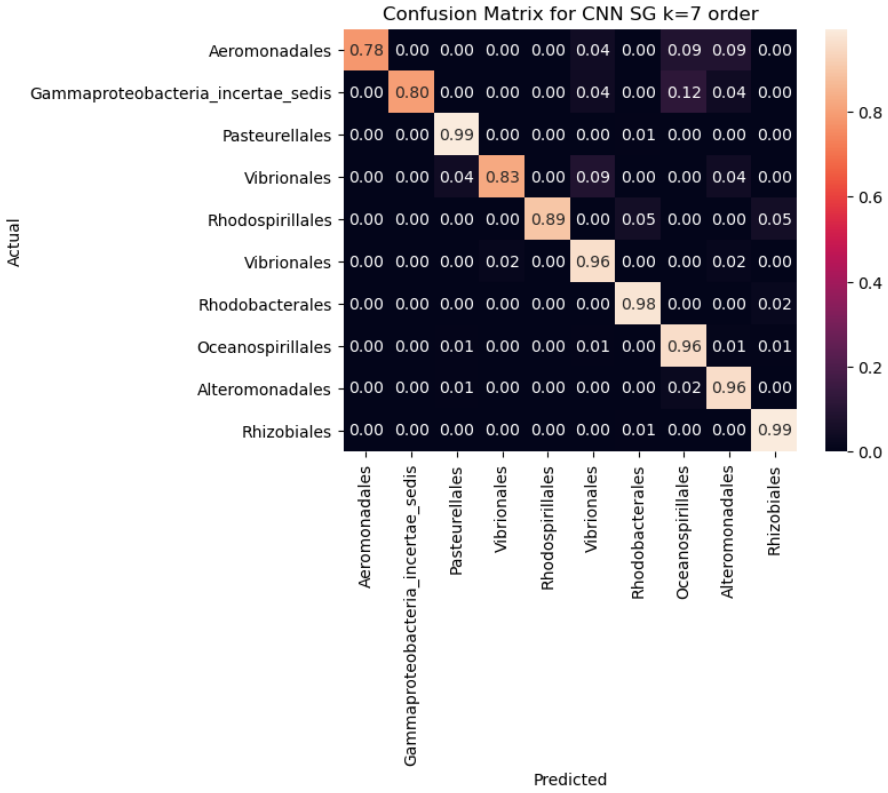

Figure A.2: Confusion matrix of the CNN model at the order level, trained with the SG dataset and 7-mers. Just the orders with more misclassifications are shown.

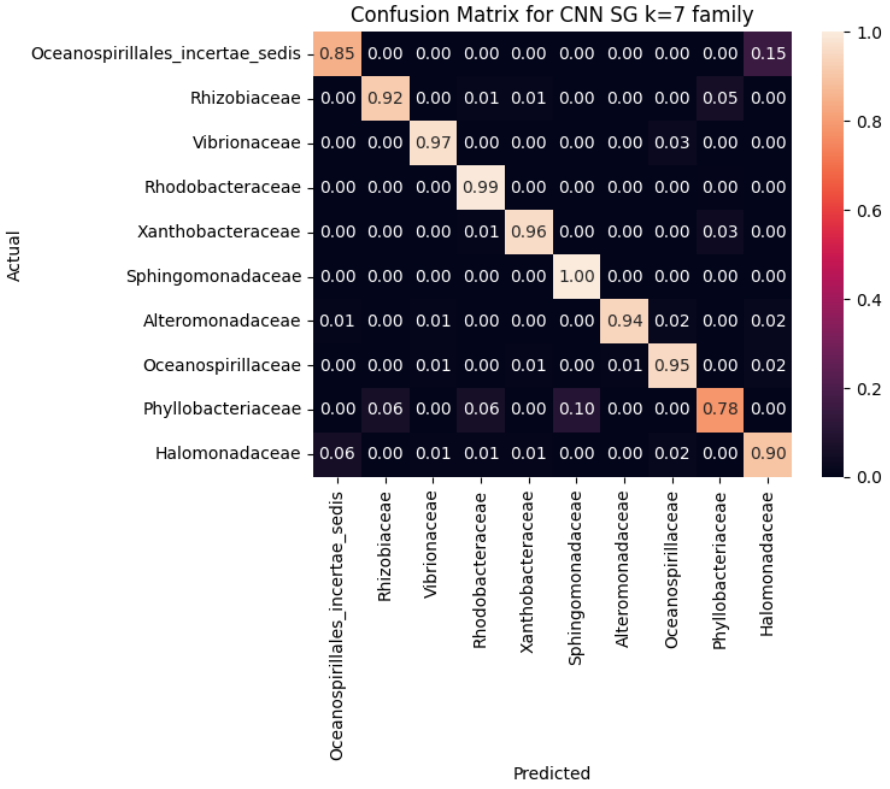

Figure A.3: Confusion matrix of the CNN model at the family level, trained with the SG dataset and 7-mers. Just the families with more misclassifications are shown.

A.2 DBN supplementary results

k-mer size	SG Accuracy			
	Class	Order	Family	Genus
3	0.808±0.012	0.466±0.012	0.343±0.013	0.124±0.006
4	0.976±0.003	0.837±0.008	0.774±0.005	0.499±0.012
5	0.993±0.002	0.934±0.005	0.896±0.003	0.708±0.006
6	0.996±0.001	0.957±0.003	0.933±0.004	0.775±0.008
7	0.997±0.001	0.964±0.003	0.944±0.005	0.810±0.005

Table A.11: Accuracy of the DBN model with SG dataset.

k-mer size	**AMP Accuracy**			
	Class	**Order**	**Family**	**Genus**
3	0.922±0.009	0.727±0.012	0.639±0.012	0.479±0.012
4	0.998±0.001	0.976±0.004	0.940±0.004	0.848±0.005
5	0.999±0.001	0.993±0.001	0.971±0.003	0.896±0.005
6	0.999±0.001	0.995±0.001	0.974±0.003	0.905±0.005
7	0.999±0.001	0.997±0.002	0.978±0.002	0.915±0.004

Table A.12: Accuracy of the DBN model with AMP dataset.

k-mer size	**SG**			
	Training time (s)			
	Class	**Order**	**Family**	**Genus**
3	12000±7000	12000±8000	12000±8000	10000±7000
4	14000±10000	12000±8000	12000±8000	20000±12000
5	19000±13000	20000±13000	13000±9000	26000±15000
6	22000±15000	24000±14000	24000±14000	10000±7000
7	10000±8000	20000±12000	20000±12000	21000±12000

Table A.13: Training time in seconds (s) for the DBN model and SG dataset.

k-mer size	**AMP**			
	Training time (s)			
	Class	**Order**	**Family**	**Genus**
3	8000±5000	12000±8000	13000±8000	8000±5000
4	17000±12000	25000±15000	12000±7000	14000±10000
5	13000±9000	16000±11000	24000±14000	23000±14000
6	23000±14000	20000±14000	17000±12000	17000±12000
7	12000±8000	19000±11000	19000±11000	19000±11000

Table A.14: Training time in seconds (s) for the DBN model and AMP dataset.

k-mer size	**SG**			
	Inference time (s)			
	Class	**Order**	**Family**	**Genus**
3	0.2±0.1	0.2±0.1	0.2±0.1	0.2±0.1
4	0.2±0.1	0.2±0.1	0.2±0.1	0.3±0.1
5	0.3±0.2	0.3±0.1	0.2±0.1	0.4±0.2
6	0.4±0.2	0.3±0.2	0.4±0.2	0.2±0.1
7	0.3±0.1	0.4±0.1	0.4±0.2	0.4±0.1

Table A.15: Inference time in seconds (s) for the DBN model and SG dataset.

k-mer size	**AMP**			
	Inference time (s)			
	Class	**Order**	**Family**	**Genus**
3	0.2±0.1	0.2±0.1	0.2±0.1	0.2±0.1
4	0.2±0.1	0.3±0.2	0.2±0.1	0.2±0.1
5	0.2±0.1	0.2±0.1	0.4±0.2	0.3±0.2
6	0.4±0.2	0.3±0.1	0.3±0.1	0.3±0.1
7	0.3±0.1	0.4±0.1	0.4±0.2	0.4±0.1

Table A.16: Inference time in seconds (s) for the DBN model and AMP dataset.

Dataset	**Precision DBN**			
	Class	**Order**	**Family**	**Genus**
SG_k3	0.823±0.010	0.562±0.018	0.427±0.029	0.164±0.014
SG_k4	0.976±0.003	0.843±0.008	0.780±0.005	0.520±0.005
SG_k5	0.993±0.002	0.935±0.005	0.898±0.003	0.720±0.008
SG_k6	0.996±0.001	0.958±0.003	0.934±0.004	0.782±0.009
SG_k7	0.997±0.001	0.964±0.003	0.945±0.005	0.815±0.005
AMP_k3	0.924±0.007	0.774±0.006	0.682±0.008	0.540±0.014
AMP_k4	0.998±0.001	0.976±0.004	0.942±0.004	0.861±0.005
AMP_k5	0.999±0.001	0.993±0.001	0.972±0.003	0.900±0.005
AMP_k6	0.999±0.001	0.995±0.001	0.975±0.003	0.907±0.004
AMP_k7	0.999±0.001	0.997±0.002	0.978±0.002	0.919±0.004

Table A.17: Precision metric for the DBN model.

Dataset	**Recall DBN**			
	Class	**Order**	**Family**	**Genus**
SG_k3	0.808±0.012	0.526±0.010	0.413±0.024	0.130±0.007
SG_k4	0.976±0.003	0.837±0.008	0.774±0.005	0.499±0.012
SG_k5	0.993±0.002	0.934±0.005	0.896±0.003	0.708±0.006
SG_k6	0.996±0.001	0.957±0.003	0.933±0.004	0.775±0.008
SG_k7	0.997±0.001	0.964±0.003	0.944±0.005	0.810±0.005
AMP_k3	0.922±0.009	0.727±0.012	0.639±0.012	0.480±0.012
AMP_k4	0.998±0.001	0.976±0.004	0.940±0.004	0.851±0.006
AMP_k5	0.999±0.001	0.993±0.001	0.971±0.003	0.896±0.005
AMP_k6	0.999±0.001	0.995±0.001	0.974±0.003	0.905±0.005
AMP_k7	0.999±0.001	0.997±0.002	0.978±0.002	0.915±0.004

Table A.18: Recall metric for the DBN model.

Dataset	**F1 Score DBN**			
	Class	**Order**	**Family**	**Genus**
SG_k3	0.790±0.015	0.461±0.010	0.311±0.021	0.115±0.007
SG_k4	0.976±0.003	0.830±0.010	0.766±0.006	0.490±0.012
SG_k5	0.993±0.002	0.933±0.005	0.895±0.003	0.707±0.006
SG_k6	0.996±0.001	0.957±0.003	0.933±0.004	0.774±0.009
SG_k7	0.997±0.001	0.964±0.003	0.944±0.005	0.809±0.005
AMP_k3	0.921±0.009	0.712±0.012	0.627±0.013	0.472±0.012
AMP_k4	0.998±0.001	0.975±0.004	0.939±0.004	0.846±0.006
AMP_k5	0.999±0.001	0.993±0.001	0.971±0.003	0.895±0.005
AMP_k6	0.999±0.001	0.995±0.001	0.974±0.003	0.904±0.004
AMP_k7	0.999±0.001	0.997±0.002	0.978±0.002	0.914±0.004

Table A.19: F1 score metric for the DBN model.

Dataset	**Area Under the Curve DBN**			
	Class	**Order**	**Family**	**Genus**
SG_k3	0.931±0.003	0.867±0.005	0.860±0.008	0.855±0.005
SG_k4	0.998±0.001	0.991±0.001	0.990±0.001	0.984±0.001
SG_k5	0.999±0.001	0.999±0.001	0.998±0.001	0.995±0.001
SG_k6	0.999±0.001	0.999±0.001	0.999±0.001	0.996±0.001
SG_k7	0.999±0.001	0.999±0.001	0.999±0.001	0.997±0.001
AMP_k3	0.986±0.002	0.981±0.002	0.975±0.002	0.979±0.001
AMP_k4	0.999±0.001	0.999±0.001	0.999±0.001	0.999±0.001
AMP_k5	0.999±0.001	0.999±0.001	0.999±0.001	0.999±0.001
AMP_k6	0.999±0.001	0.999±0.001	0.999±0.001	0.999±0.001
AMP_k7	0.999±0.001	0.999±0.001	0.999±0.001	0.999±0.001

Table A.20: Area under the curve metric for the DBN model.

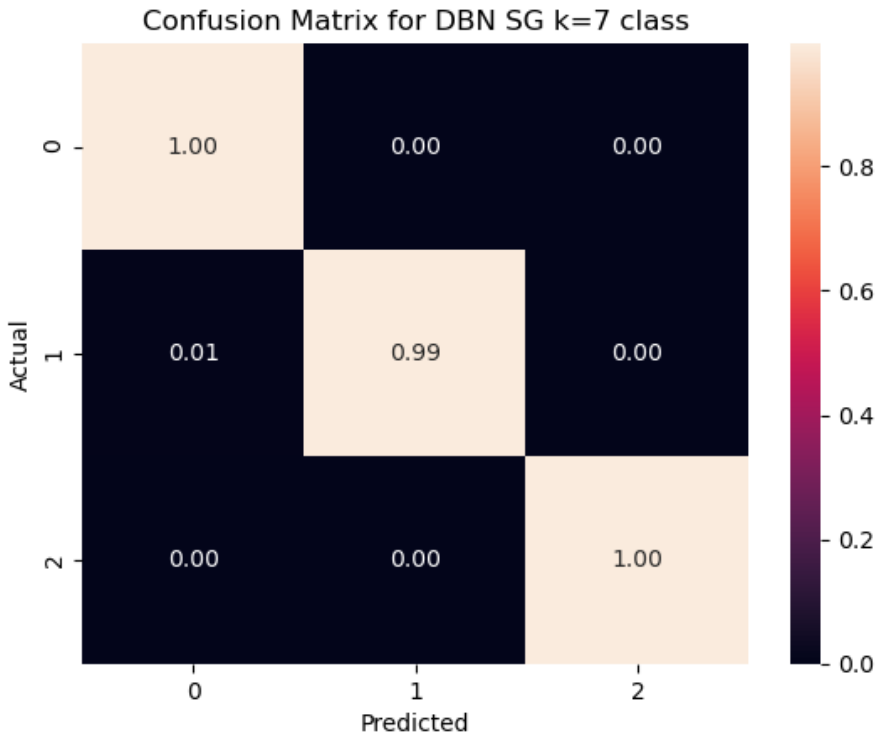

Figure A.4: Confusion matrix of the DBN model at the class level, trained with the SG dataset and 7-mers.

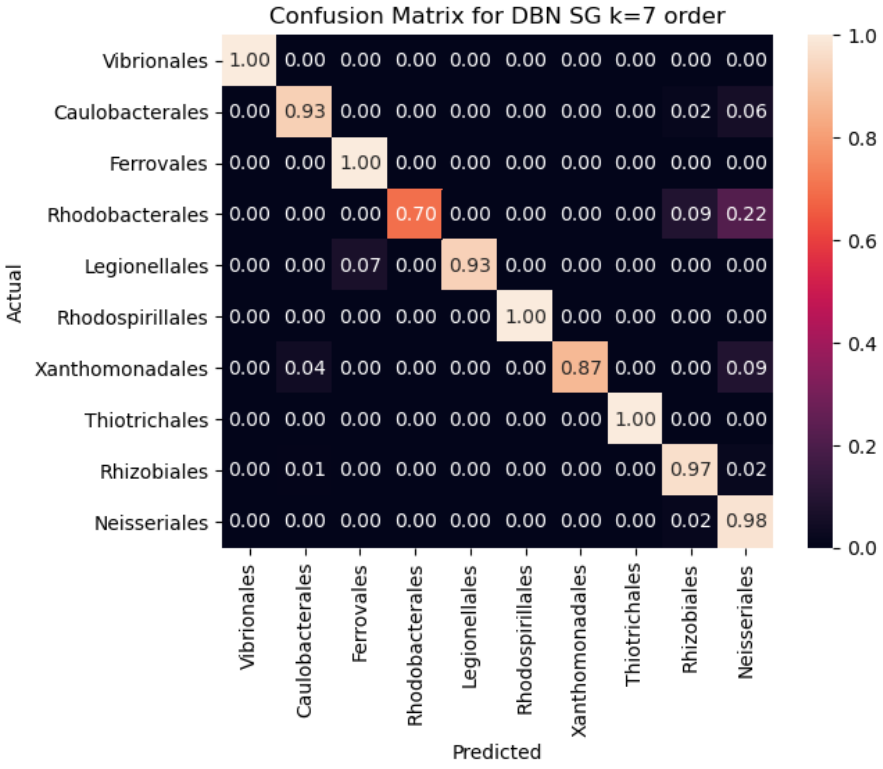

Figure A.5: Confusion matrix of the DBN model at the order level, trained with the SG dataset and 7-mers. Just the orders with more misclassifications are shown.

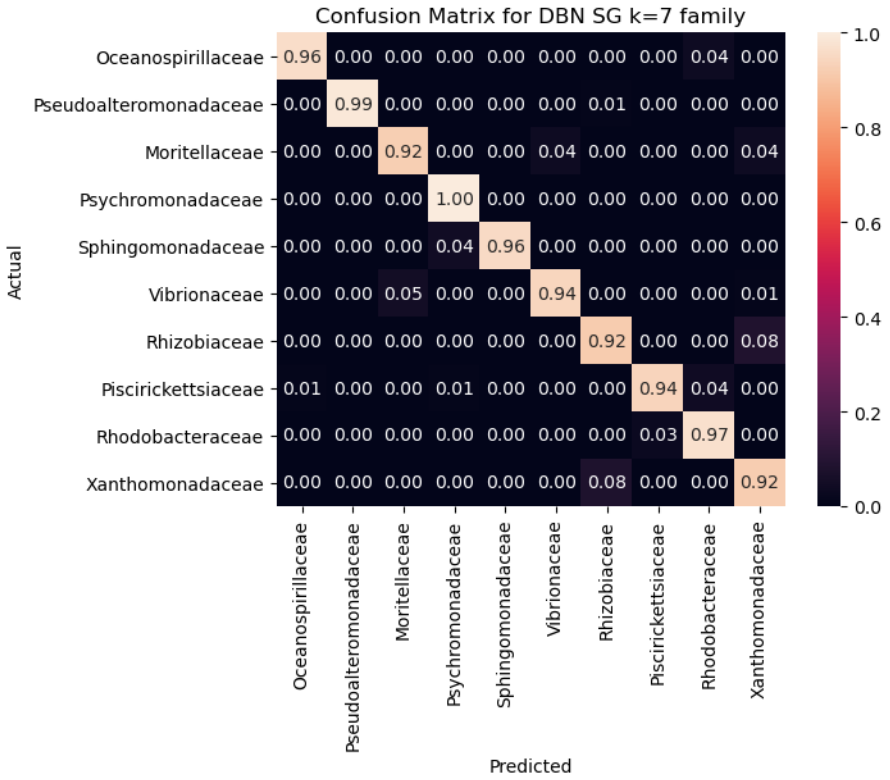

Figure A.6: Confusion matrix of the DBN model at the family level, trained with the SG dataset and 7-mers. Just the families with more misclassifications are shown.

A.3 XGBoost supplementary results

k-mer size	**SG Accuracy**			
	Class	**Order**	**Family**	**Genus**
3	0.820±0.006	0.502±0.006	0.421±0.007	0.253±0.005
4	0.939±0.004	0.755±0.008	0.711±0.007	0.557±0.006
5	0.985±0.002	0.904±0.005	0.890±0.004	0.768±0.008
6	0.993±0.001	0.946±0.005	0.938±0.004	0.838±0.007
7	0.994±0.001	0.959±0.003	0.947±0.004	0.838±0.009

Table A.21: Accuracy of the XGBoost model with SG dataset.

k-mer size	**AMP Accuracy**			
	Class	**Order**	**Family**	**Genus**
3	0.952±0.004	0.812±0.006	0.721±0.006	0.548±0.010
4	0.997±0.001	0.970±0.005	0.936±0.004	0.825±0.005
5	0.999±0.001	0.990±0.001	0.970±0.002	0.883±0.005
6	0.999±0.001	0.994±0.001	0.976±0.002	0.901±0.003
7	0.999±0.001	0.994±0.002	0.976±0.002	0.902±0.004

Table A.22: Accuracy of the XGBoost model with AMP dataset.

k-mer size	**SG**			
	Training time (s)			
	Class	**Order**	**Family**	**Genus**
3	1.51±0.07	8.9±0.1	16.4±0.2	40.1±0.3
4	3.8±0.3	26±2	41.5±0.2	101.7±0.3
5	11.4±0.2	74.8±0.3	137.2±0.9	342±1
6	40.3±0.3	270±2	495±3	1254±3
7	144.0±0.9	944±2	1750±2	4609±255

Table A.23: Training time in seconds (s) for the XGBoost model and SG dataset.

k-mer size	**AMP**			
	Training time (s)			
	Class	**Order**	**Family**	**Genus**
3	1.8±0.3	9.9±0.6	15.2±0.3	37.0±0.4
4	3.3±0.1	17.3±0.2	30.6±0.1	70.5±0.7
5	8.6±0.1	46.8±0.5	89±1	208.4±0.9
6	28.0±0.3	154±2	296±3	705±2
7	101±1	530±2	1037±4	2484±6

Table A.24: Training time in seconds (s) for the XGBoost model and AMP dataset.

k-mer size	**SG**			
	Inference time (s)			
	Class	**Order**	**Family**	**Genus**
3	0.004±0.001	0.012±0.001	0.028±0.006	0.10±0.01
4	0.006±0.003	0.019±0.005	0.041±0.007	0.11±0.01
5	0.011±0.004	0.03±0.01	0.04±0.01	0.13±0.02
6	0.02±0.01	0.05±0.02	0.06±0.02	0.15±0.02
7	0.07±0.03	0.09±0.03	0.13±0.04	0.20±0.04

Table A.25: Inference time in seconds (s) for the XGBoost model and SG dataset.

k-mer size	**AMP**			
	Inference time (s)			
	Class	**Order**	**Family**	**Genus**
3	0.006±0.002	0.016±0.005	0.03±0.01	0.10±0.02
4	0.008±0.003	0.015±0.004	0.019±0.002	0.06±0.01
5	0.008±0.005	0.015±0.006	0.019±0.004	0.040±0.006
6	0.02±0.01	0.04±0.01	0.04±0.01	0.06±0.02
7	0.09±0.06	0.06±0.03	0.06±0.01	0.09±0.01

Table A.26: Inference time in seconds (s) for the XGBoost model and AMP dataset.

Dataset	**Precision XGBoost**			
	Class	**Order**	**Family**	**Genus**
SG_k3	0.821±0.008	0.522±0.008	0.434±0.014	0.258±0.006
SG_k4	0.940±0.004	0.789±0.008	0.737±0.007	0.561±0.007
SG_k5	0.985±0.002	0.912±0.004	0.894±0.005	0.772±0.008
SG_k6	0.993±0.001	0.948±0.005	0.940±0.003	0.843±0.008
SG_k7	0.994±0.001	0.960±0.003	0.948±0.004	0.842±0.008
AMP_k3	0.952±0.004	0.819±0.007	0.725±0.006	0.553±0.011
AMP_k4	0.997±0.001	0.971±0.005	0.937±0.004	0.828±0.005
AMP_k5	0.999±0.001	0.990±0.001	0.970±0.002	0.885±0.005
AMP_k6	0.999±0.001	0.994±0.001	0.976±0.002	0.904±0.003
AMP_k7	0.999±0.001	0.994±0.002	0.977±0.002	0.905±0.003

Table A.27: Precision metric for the XGBoost model.

Dataset	**Recall XGBoost**			
	Class	**Order**	**Family**	**Genus**
SG_k3	0.820±0.006	0.502±0.006	0.421±0.007	0.253±0.005
SG_k4	0.939±0.004	0.755±0.008	0.711±0.007	0.557±0.006
SG_k5	0.985±0.002	0.904±0.005	0.890±0.004	0.768±0.008
SG_k6	0.993±0.001	0.946±0.005	0.938±0.004	0.838±0.007
SG_k7	0.994±0.001	0.959±0.003	0.947±0.004	0.838±0.009
AMP_k3	0.952±0.004	0.812±0.006	0.721±0.006	0.548±0.010
AMP_k4	0.997±0.001	0.970±0.005	0.936±0.004	0.825±0.005
AMP_k5	0.999±0.001	0.990±0.001	0.970±0.002	0.883±0.005
AMP_k6	0.999±0.001	0.994±0.001	0.976±0.002	0.901±0.003
AMP_k7	0.999±0.001	0.994±0.002	0.976±0.002	0.902±0.004

Table A.28: Recall metric for the XGBoost model.

Dataset	**F1 Score XGBoost**			
	Class	**Order**	**Family**	**Genus**
SG_k3	0.809±0.007	0.472±0.007	0.386±0.007	0.253±0.006
SG_k4	0.937±0.004	0.742±0.008	0.699±0.008	0.555±0.006
SG_k5	0.985±0.002	0.902±0.005	0.889±0.004	0.767±0.008
SG_k6	0.993±0.001	0.945±0.005	0.938±0.004	0.838±0.007
SG_k7	0.994±0.001	0.958±0.003	0.946±0.004	0.838±0.008
AMP_k3	0.952±0.004	0.807±0.006	0.716±0.006	0.547±0.010
AMP_k4	0.997±0.001	0.970±0.005	0.935±0.004	0.824±0.005
AMP_k5	0.999±0.001	0.990±0.001	0.970±0.002	0.882±0.005
AMP_k6	0.999±0.001	0.994±0.001	0.976±0.002	0.901±0.003
AMP_k7	0.999±0.001	0.994±0.002	0.976±0.002	0.902±0.004

Table A.29: F1 score metric for the XGBoost model.

Dataset	**Area Under the Curve XGBoost**			
	Class	**Order**	**Family**	**Genus**
SG_k3	0.923±0.004	0.897±0.004	0.900±0.003	0.901±0.002
SG_k4	0.992±0.001	0.980±0.001	0.982±0.001	0.977±0.001
SG_k5	0.999±0.001	0.997±0.001	0.997±0.001	0.994±0.001
SG_k6	0.999±0.001	0.999±0.001	0.999±0.001	0.997±0.001
SG_k7	0.999±0.001	0.999±0.001	0.999±0.001	0.997±0.001
AMP_k3	0.994±0.001	0.987±0.001	0.984±0.001	0.982±0.001
AMP_k4	0.999±0.001	0.999±0.001	0.999±0.001	0.997±0.001
AMP_k5	0.999±0.001	0.999±0.001	0.999±0.001	0.998±0.001
AMP_k6	0.999±0.001	0.999±0.001	0.999±0.001	0.999±0.001
AMP_k7	0.999±0.001	0.999±0.001	0.999±0.001	0.999±0.001

Table A.30: Area under the curve metric for the XGBoost model.

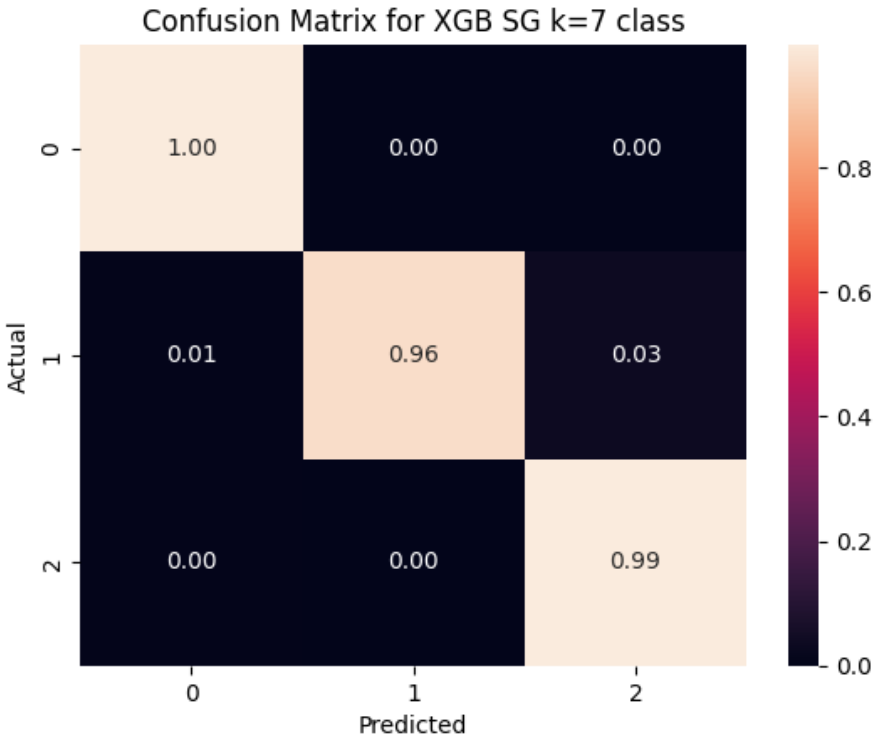

Figure A.7: Confusion matrix of the XGBoost model at the class level, trained with the SG dataset and 7-mers.

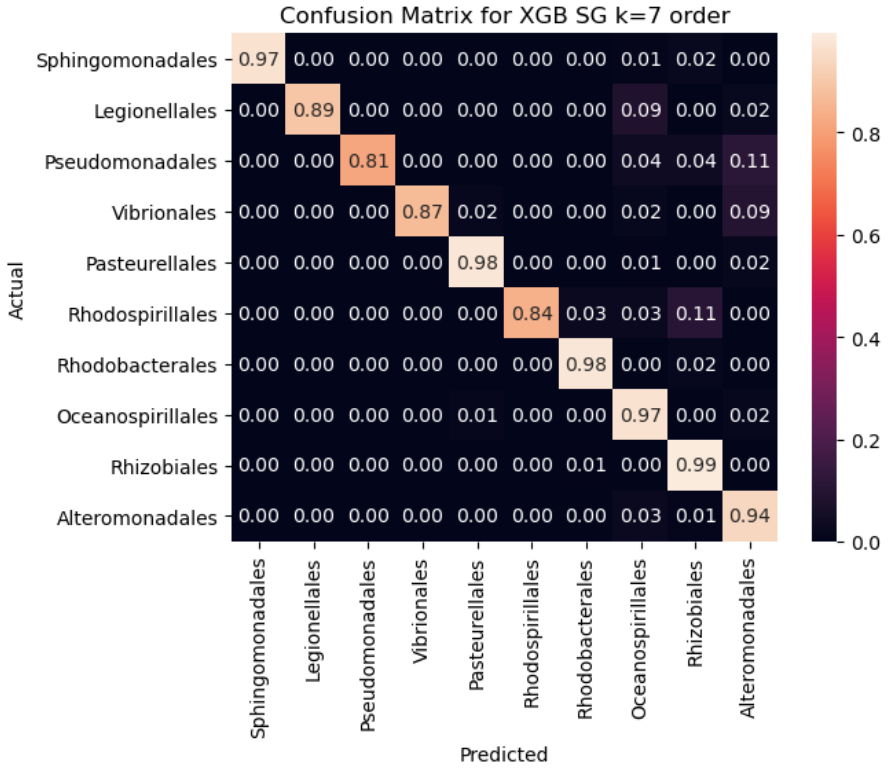

Figure A.8: Confusion matrix of the XGBoost model at the order level, trained with the SG dataset and 7-mers. Just the orders with more misclassifications are shown.

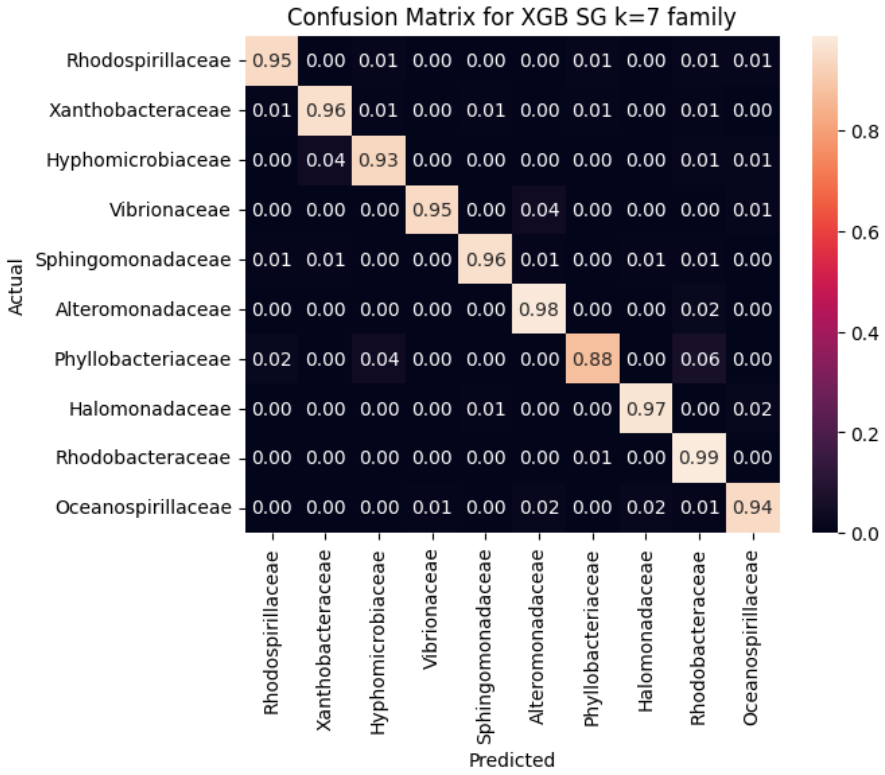

Figure A.9: Confusion matrix of the XGBoost model at the family level, trained with the SG dataset and 7-mers. Just the families with more misclassifications are shown.