PhageLysin: Phage Endolysin Finder

Tiago Baptista[1], Hugo Oliveira[2], and Óscar Dias[2]

[1] University of Minho, Portugal
[2] University of Minho, Center of Biological Engineering (CEB), Portugal

Abstract. Enzybiotics are emerging as a promising alternative to antibiotics, with endolysins showing great potential, due to their muralytic activity, which enables them to lyse bacteria. A bioinformatic tool to identify and classify endolysins in new phage genomes and metagenomic sequencing data is still a gap to be filled, thus this work aims to establish steps in that direction. In this project, we focus on building an endolysin dataset to train machine learning models and achieve endolysins prediction. Several models were used with different machine learning algorithms such as Logistic Regression, Random Forest, Support Vector Machine, Decision Tree and Artificial Neural Network. In order to improve the performance of these models, we performed hyperparameter tuning. These models were evaluated taking into account different metrics, which resulted in the Random Forest model having the best performance, scoring 91% in all metrics. However, these models need to be improved in the future, despite results being satisfactory, we are still far from successfully developing a consistent tool for endolysins prediction.

1 Introduction

1.1 Aims

Bacterial multidrug resistance to antibiotics is one of the most concerning problems faced in modern days, however, new alternative therapies are being developed. The potential use of enzybiotics, a relatively new term that defines therapies using phage-encoded enzymes, normally phage lytic enzymes, is rising as arguably the best alternative to regular antibiotics. One of those enzymes is the endolysin, utilized by phages to degrade the cell wall from within. There are also several studies that have shown that recombinant endolysins can also be applied to degrade cell wall from the outside, thus confirming their antibacterial properties [1] [3].

This project aims at building a comprehensive dataset of phage endolysins and exploring ML approaches to improve the accuracy of predictions i.e. classify endolysins. The ultimate goal, probably outside of the scope of this project, would be the development of a machine learning (ML) tool to provide a quick and reliable way to identify endolysins in new phage genomes, but also on metagenomic sequencing data.

1.2 Phages and endolysins

Phages represent the most plentiful and varied biological entities on the planet and are thought to exist in virtually every imaginable environment. It is estimated that there are 10^{31} phages, outnumbering their bacterial hosts by an estimated tenfold [6] [14] [16]. Phages exhibit remarkable diversity with a wide range of genome sizes from as low as 2,435 bp to as high as >540 kb [6].

Phage classification is based on various characters that describe the virus and allow to differentiate one from another. Characters include phage morphology and genome composition, area amongst the most important ones. Even though all characters are important in determining taxonomy, sequence comparisons using both pairwise sequence similarity and phylogenetic relationships are the main sets of characters to define and distinguish virus taxa [11]. Furthermore, phage taxonomy is a complex subject and is regularly under revision by the International Committee on Taxonomy of Viruses (ICTV) [21]. A recent ICTV taxonomy update constituted the class *Caudoviricetes* as one of the most important ones, encompassing all tailed bacterial and archaeal viruses with icosahedral capsids and a double stranded DNA (dsDNA) genome [18].

Phage replication first requires infection of the bacteria so they must bind to the bacterium surface and subsequently resort to a spike shaped complex of proteins at the bottom of the baseplate of phage's structure that punctures into the bacterial host, allowing a release of phage's genetic material into the host's intracellular environment. After that, phages can undergo one of two possible life cycles: lytic or lysogenic. The lytic cycle is followed by strictly lytic phages (also termed virulent phages) and consists in the seizure of bacterial host's metabolic machinery, which is first redirected to replication of phage's genome to create multiple copies of its own genetic material. Then, translation occurs and the synthesis of viral proteins and enzymes is achieved. Among them, some are used to assemble capsid and tails of new phages and package their genome, while enzymes such as holins and endolysins allow bacterium lysis and newly assembled phages release. In turn, the lysogenic cycle exhibited particularly by temperate phages requires the integration of the viral genome within the host's genome. This lysogenic bacteria has an integrated prophage within its genome, that can remain dormant through numerous bacterial cell divisions unless triggered by stress or cellular damage in the bacterial host. Upon activation by these factors, the phage cycle shifts towards replication through a lytic pathway, after excision from the bacterial genome, culminating in the release of the new phage particles [9] [16].

The holin-endolysin lytic system, also known as the lambda paradigm, is the most common in dsDNA phages. As depicted in Fig. 1, this system consists of endolysin accumulation in the intracellular space due to the bacterial inner membrane. This accumulation goes on until the holin, a small hydrophobic membrane-spanning protein, after being expressed at a genetically programmed time, had accumulated in clusters producing homo-oligomeric pores in the inner membrane. Holin hole formation triggers the activation of the endolysin by granting its access to the peptidoglycan (PG) (or murein), a heteropolymer con-

sisting of disaccharide peptide repeat units linked by glycosidic bonds that form glycan strands. These strands are cross-linked through pentapeptides by 4–3 and 3–3 linkages. Endolysin muralytic activity i.e. its capability to degrade the PG, results in an unsustainable internal osmotic pressure, leading to cell lysis and progeny release. [4] [14]. Endolysins classification based on their lytic activity can be categorised into glycosidases, amidases and endopeptidases [1] [14] as shown in Fig. 1.

Endolysins unique ability to rapidly kill bacteria in a species specific manner puts them as promising antibacterial and biocontrol agents with applications in fermentations, food preservation, biotechnology, and medicine. Studies regarding the capacity of endolysins to lyse bacteria when externally added have already been performed. In the case of Gram-positive bacteria, cell lysis was already achieved *in vitro*, leading to the complete death of a streptococcus culture in a few seconds [14]. Other *in vivo* studies have shown endolysins efficacy against a variety of Gram-positive bacteria such as PlyC, C1, ClyR, Cpl-1, ClyV, and ClyJ [1]. Regarding Gram-negative bacteria, the exogenous action of endolysins is still restricted due to the presence of the impermeable outer membrane, constituting one of the most important challenges in endolysin therapy, however in recent years molecular engineering approaches have increased the applicability of endolysins in targeting Gram-negative bacteria [1] [14].

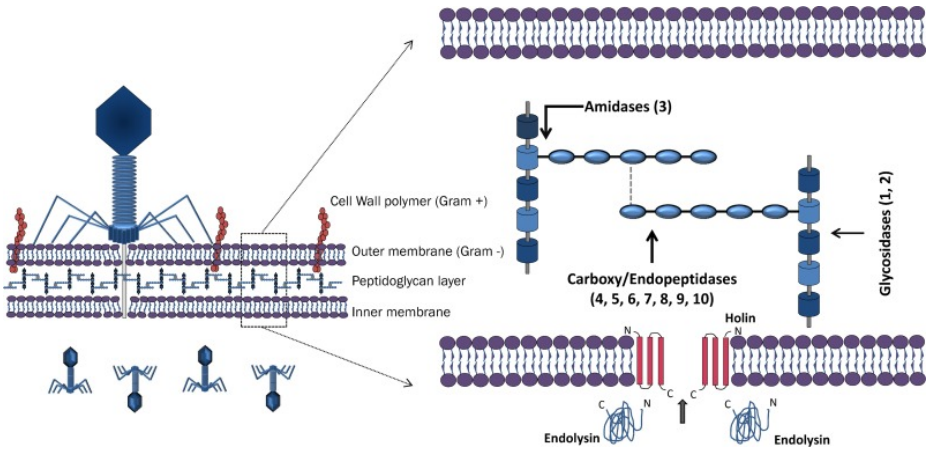

Fig. 1: Schematic representation of phage endolysins access to the PG through the holin-endolysin lytic system, alongside with a generic PG structure illustration of endolysins cleavage sites [14].

1.3 Phage endolysins databases

Phage lytic proteins databases containing endolysins are available, namely phiBIOTICS [10], EnzyBase (renamed to EnzyBase2) [23], GMEnzy [22], PhaLP [5]

and PhalydDB [8]. phiBIOTICS, EnzyBase2 and GMEnzy data are collected manually so they naturally have few entries. Oppositely, PhaLP and PhalydDB are comprehensive databases with PhaLP being automatically updated with each new UniProt release, thus containing a significantly larger amount of entries [5] [8]. PhaLP allows database querying via MySQL [5]. In contrast, phiBIOTICS, EnzyBase2 and GMEnzy are not optimised for this purpose, so manual retrieval of information, use of APIs, or web scraping may be necessary to query these databases. PhalydDB is a recent project, and according to a data sharing statement by Gou et al. [8], it is necessary to request authors access to their data.

Even though it is not a specialized database for phage lytic proteins, endolysins can be queried from the National Center for Biotechnology Information (NCBI)'s Protein database [13], for example using the Biopython package, thus providing an alternative way to build or complete the endolysins database.

1.4 Machine learning approaches to predict endolysins

ML is a field in computer science focused on giving computers the ability to acquire knowledge and improve their performance on specific tasks, all without explicit programming. This approach has roots in the rise of artificial intelligence in the 1950s, and prioritizes practical applications, especially those related to prediction and optimisation. The learning process involves computers being exposed to data, which they analyze to improve their performance over time. In essence, this "experience" translates to fitting the data to identify patterns and make predictions [2].

ML approaches provide a promising way not only to predict phage virion proteins but also their functions. Since we want to predict if a certain sequence is an endolysin or not, we have a classification problem.

Nami et al. [12] conducted a review of 10 different ML models for the previous purposes. The list of the models, alongside the respective methods and features, is presented in table 1. Table 1 also shows diversity regarding ML methods and features, allowing flexibility when choosing the ML model(s) for the scope of this project. Nevertheless, model performances must be evaluated in the context of our work, to allow for a weighted decision of a model in detriment of another.

In a recent work, Vieira et al. [19] developed the first ML tool to identify phage depolymerases. This tool is based on two algorithms, Support Vector Machines (SVMs) and Artificial Neural Networks (ANNs) that were used to train the machine learning models. This groundbreaking study shows great progress in ML tools to identify phage lytic enzymes providing valuable insights for the future development of an ML tool to predict endolysins.

In the previous studies, algorithms such as SVM, ANN, and Random Forest (RF) were used. SVM can provide high accuracy and is based on the concept of maximizing the minimum distance from hyperplane to the nearest sample point. Even if data is not linearly separable in base feature space, they can perform well, with an appropriate kernel. Accuracy and performance are dependent on the number of training cycle, but not on the data size. Number of features does not affect complexity. Robust when trained with high dimensional data

Table 1: ML models and respective methods and features [12].

Predictor	**Method**
ANN	"ACC, protein isoelectric Points" + ANN
Naïve Bayes	"ACC, DPC" + CFS + Naïve Bayes
PVPred	g-gap DPC + ANOVA+SVM
PhagePred	g-gap DPC + ANOVA + Multinomial Naïve Bayes + ANN
PVP-SVM	"AAC, ATC, CTD, DPC, PCP" + RF-based feature selection + SVM
SVM-based	g-gap DPC + "ANOVA, mRMR" + SVM + ANN
Ensemble RF	"CTD, bi-profile Bayes, PseAAC, PSSM" + Relief + RF + ANN
Pred-BVP-Unb	CT, SAAC, bi-PSSM+SVM + ANN
PVPred-SCM	DPC + SCM + ANN
Meta-iPVP	Probabilistic feature+SVM + ANN

SCM scoring card method, *SVM* support vector machine, *AAC* amino acid composition, *ATC* atomic composition, *bi-PSSM* bi-profile position specific scoring matrix, *CTD* chain-transition-distribution, *CT* composition and translation, *DPC* dipeptide composition, *GDPC* g-gap dipeptide composition, *PCP* physicochemical properties, *SAAC* split amino acid composition.

and generalise well. Performance is dependent on the chosen parameters. ANNs are computational devices based on human brain neuronal structure, processing method and learning ability. More suited to non-linear and dynamic relationships, providing an alternative to other algorithms that assume normality, linearity, and variable independence. Multi-Layer Perceptron (MLP) is the most used ANN classifier and is robust to non relevant input and noise. Hard to train and to determine the right size of the hidden layer. Performance depends on the chosen parameter values. RF is an ensemble method that trains a number of decision trees and returns the major class over all the trees. Known for being fast, scalable, robust to noise, no overfitting, easy to interpret and visualize. An increase in the number of trees has a significant impact on the computation time. Decision Tree (DT) and Logistic Regression (LogR) are another two popular algorithms for classification problems. DT rely on the divide and conquer approach, performing well when few relevant attributes are present and the opposite when there are complex interactions. Error propagation through trees is one of its biggest problems, which is aggravated with an increase of the number of classes. It is non-parametric, so outliers have no impact on performance and it can deal with linearly inseparable data. Relatively easy to interpret and explain and can handle interactions between features. Have no problem with redundant attributes, generalise well, are robust to noise, have high performance, and require relatively little computational power, however have trouble handling high dimensional data. Without proper pruning it tends to over fit, hence it is why RF ensemble model was developed. LogR is a statistical model that fits a logistic curve to the dataset. It is applied in cases where the target variable is dichotomous, which is our case. It allows a probabilistic interpretation and easy model updating to add new data. Classification thresholds can be easily adjusted, since

it returns a probability. It requires a large amount of data to achieve robust and stable results [17].

Besides the algorithms, it is also necessary to obtain the features to build the models. Sequeira et. al [15] developed Propythia, a python package capable of generating descriptors for biological sequences, including protein sequences. These descriptors can be calculated/obtained from the sequences, and represent their characteristics. Moreover, these descriptors provide a simple approach to build the models' features.

2 Materials and Methods

Our main objective consisted on developing a ML pipeline for endolysin prediction using the scikit-learn package. Scikit-learn allows data preprocessing and implementation of an efficient and functional machine learning model. Fig. 2 shows a schematic representation of the ML pipeline.

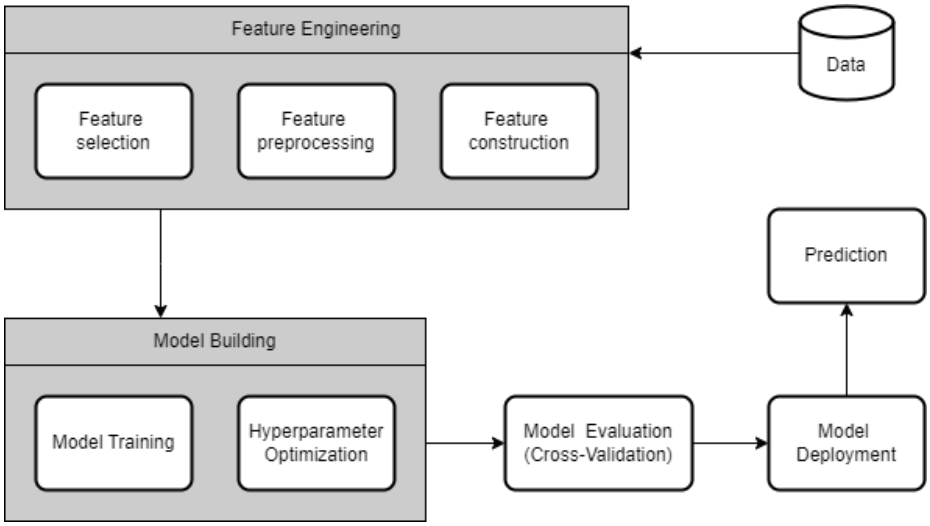

Fig. 2: Machine Learning pipeline developed in this work.

Firstly, we needed to obtain the data to build the positive and negative datasets to train ML models. To build the dataset of phage endolysins that would be further used as the positive dataset, we obtained endolysin sequences by querying the NCBI protein database, instead of retrieving them from the previously referred phage lytic protein databases. We used the query *endolysin[Protein Name] AND txid28883[Organism]* to select all proteins named endolysin from phages belonging to the *Caudoviricetes* class. We queried the database on 27/05/2024 retrieving 9,495 sequences in a FASTA file. To build the negative

dataset, we selected the structural proteins from Phage Artificial Neural Networks (PhANNs), a tool to predict and classify phage Open Reading Frames (ORFs) of structural proteins. Moreover, a total of 547,911 sequences were selected and gathered in a FASTA file. After obtaining the sequences we performed a filtering by removing duplicates and running the remaining sequences through CD-HIT. CD-HIT is a widely used program for clustering biological sequences to reduce sequence redundancy and improve the performance of other sequence analyses [7]. We used CD-HIT default parameters, including a sequence identity threshold of 0.9, i.e. it will cluster sequences that share at least 90% identity. This filtering resulted in a reduction of the number of sequences to 2,758 and 57,389 for positive and negative datasets, respectively. Looking at the number of sequences of both datasets it is clear that the negative dataset is larger than the positive one, so we performed a random selection of sequences from the negative dataset, reducing it to the same size of the positive one. We also performed a random selection to obtain another negative dataset with double the size of the positive one, with the intent to represent better the ratio of endolysins/structural proteins in phage genomes. After construction and preprocessing of the positive and negative datasets, we had them featuring three columns, namely *accession_id* containing the accession ids of sequences, *sequence* containing the complete protein sequences and *target* containing the identifiers *endolysin* and *not_endolysin* for the positive and negative datasets, respectively. The next step consisted of generating numeric protein descriptors from protein sequences, since ML models can only be trained on numeric data. To achieve that, we applied the *get_all_physicochemical* method of Propythia package to generate all its physicochemical protein descriptors, listed on table 2. Finally, to finish the construction of the dataset we merged the positive and negative datasets, defined *accession_id* column as dataframe index, removed the non-numeric *sequence* column, converted the nominal variable *target* into a binary variable with 0 and 1 representing, respectively, *not_endolysin* and *endolysin*, and saved it as a CSV file.

Table 2: Propythia physicochemical protein descriptors.

Descriptor	**Description**
length	length
charge	charge
chargedensity	charge density
formula	calculates number of carbon, hydrogen, nitrogen, oxygen and sulfur of the protein sequence
bond	total number of hydrogen, single, double and aromatic bonds
mw	molecular weight
gravy	gravy from a sequence (accordingly to biopython)
aromacity	aromacity (accordingly to biopython)
isoeletric point	isoelectric (accordingly to biopython)
instability index	instability (accordingly to biopython)
secondary structure	fraction of amino acids that tend to be in helix, turn or sheet
molar extinction coefficient	value of reduced cysteins and oxidized (with disulfid bridges)
flexibility	flexibility according to Vihinen et al., 1994 [20] (return proteinsequencelenght-9 values) from biopython
aliphatic index	aliphatic index of sequence (1 value) from modlamp
boman index	boman index of sequence (1 value) from modlamp
hydrophobic ratio	hydrophobic ratio from modlamp

The first step of supervised learning consists of dividing both datasets into X and y sets, with X being the dataframe that contains the model that will be trained and y the column vector that contains the prediction variable, in this case *target*. The X and y sets were also split into training and test sets, with 70% for the training set and the remaining 30% for the test set.

The second step aims at creating an instance of the learning algorithm that would be used to build the predictive model. Since the variable we want to predict is nominal, we have a classification problem, so we used scikit-learn *LogisticRegression*, *RandomForestClassifier*, *SVC*, *DecisionTreeClassifier* and *MLPClassifier* classes as estimators. Then we performed a univariate selection of the features using *SelectKBest* class, which selects the features with the best scores based on univariate statistical tests.

The third step involves testing different hyperparameters, for the different algorithms used in the models. Table 3 represents the hyperparameters for each algorithm, for the datasets with equal and double ratio of negatives/positives, respectively. To perform hyperparameters testing, we used the scikit-learn *Pipeline* class, which groups the previous feature selection steps and the estimator we want to use for each model algorithm. It is then used in the scikit-learn *GridSearchCV* class, together with the sets of hyperparameters we want to test. The result of this step is the best pipeline, with the respective best hyperparameter values.

Table 3: Different hyperparameters tested in the different algorithms trained with the datasets with equal (top table) and double (bottom table) ratio of negatives/positives. The best performing hyperparameters, resulting from the *GridSearchCV* hyperparameter optimisation, are highlighted in bold.

	LogR	**RF**	**SVM**	**DT**	**ANN**
Selector K	25 ('all'), 20, **15**	**25 ('all')**, 20, 15	25 ('all'), **20**, 15	25 ('all'), **20**, 15	25 ('all'), 20, **15**
Penalty	l1, **l2**, elasticnet	None	None	None	None
Estimator Number	None	10, 50, **100**	None	None	None
Estimator C	None	None	0.01, 0.1, **1.0**	None	None
Criterion	None	None	None	**gini**, entropy, log_loss	None
Hidden Layer Sizes	None	None	None	None	**(10,)**, (50,), (100,)

	LogR	**RF**	**SVM**	**DT**	**ANN**
Selector K	25 ('all'), **20**, 15	**25 ('all')**, 20, 15	25 ('all'), **20**, 15	25 ('all'), **20**, 15	**25 ('all')**, 20, 15
Penalty	l1, **l2**, elasticnet	None	None	None	None
Estimator Number	None	10, **50**, 100	None	None	None
Estimator C	None	None	0.01, 0.1, **1.0**	None	None
Criterion	None	None	None	gini, **entropy**, log_loss	None
Hidden Layer Sizes	None	None	None	None	(10,), (50,), **(100,)**

The fourth step is the evaluation of the model, through cross-validation, with the hyperparameters already optimised. Model performance was evaluated through different scoring metrics, including accuracy, precision, recall and F1 score. The scikit-learn *cross-validate* function taking as arguments the pipeline, the dataset and the scoring metrics, was used to perform a 5-fold cross-validation. Briefly, in this case, this function divided data into 5 subsets (folds), using in

each iteration, 4 folds to train the model and 1 as test set to evaluate the model. After evaluating the model, it must be retrained with the complete dataset, before being used to make predictions for external test datasets. Then, the trained model is used to predict the output variable (y) for the test set, not used during the training phase. We further evaluated model performance by calculating the previous metrics again, this time using scikit-learn *accuracy_ score*, *precision_ score*, *recall_ score* and *f1_ score* methods. In addition, we built the model confusion matrix, which contains the number of true positive, true negative, false positive, and false negative values. We also plotted the Receiver Operating Characteristic (ROC) curve and its Area Under the Curve (AUC), which plots the True Positive Rate (TPR), also known as recall, which represents the proportion of positive examples that were correctly classified as positive, versus the False Positive Rate (FPR), which represents the proportion of negative examples that were incorrectly classified as positive. TPR and FPR values range from 0 to 1, with an ideal ROC curve being a straight line rising from the bottom left corner (0 FPR, 0 TPR) to the top left corner (0 FPR, 1 TPR) and then diagonally across to the top right corner (1 FPR, 1 TPR).

The fifth and final step is saving the model in pickle format to allow for future use.

3 Results and Discussion

To predict endolysins, algorithms such as LogR, RF, SVM, DT and ANN were used. These algorithms allowed the creation of several models to predict the *target* variable values (endolysin and not endolysin) for new data. As described previously, four scoring metrics were used to evaluate the models' performance. Results are shown in table 4, for each algorithm trained with the datasets with equal and double ratio of negatives/positives.

Table 4: Score metrics obtained for each model trained with the datasets with equal (left table) and double (right table) ratio of negatives/positives.

	LogR	**RF**	**SVM**	**DT**	**ANN**
Accuracy	83%	91%	81%	84%	76%
Precision	81%	91%	81%	85%	71%
Recall	87%	91%	81%	84%	88%
F1 Score	84%	91%	81%	84%	79%

	LogR	**RF**	**SVM**	**DT**	**ANN**
Accuracy	85%	92%	83%	87%	77%
Precision	80%	91%	79%	79%	61%
Recall	74%	86%	65%	82%	85%
F1 Score	77%	88%	71%	81%	71%

Analysing the overall performance of the models for both datasets in table 4, we conclude that doubling the negative/positive ratio resulted in a drop in the performance for all models in all metrics except accuracy. Despite that, accuracy score should not be significantly considered for the 'double' dataset since it is unbalanced, and this metric is heavily dependent on the number of values for each decision class. This drop in performance is probably related to the significant

imbalance in the number of cases since most ML algorithms perform better when the classes are balanced, which may result in models' tendency to be biased towards the majority class, leading to poor performance on the minority class. This would also explain the increase in accuracy, regarding that algorithms may focus more on correctly predicting the majority class, with the cost being the reduction of other metrics score for the minority class. Another factor could be the occurrence of underfitting for the minority class since models might struggle to learn the decision boundary effectively, as they encounter fewer examples of this class. Contrarily, overfitting to the majority class may also occur, since it could contain redundant or less informative samples, however, it seems less likely considering that in principle *CD-HIT* removed those samples. Despite that, model complexity increases with the increase in the number of input features, which may lead to overfitting. Some Propythia descriptors could be redundant and may contribute to that.

Since results with the 'double' dataset were considerably worse, we will only analyse models' performance with the 'equal' dataset. Table 4 shows that metrics scores differ significantly across models with RF having the best performance scoring 91% in all metrics. Oppositely, ANN performed the worst in all metrics, except recall. In fact, ANN had the second best recall across all models with 88% score, contrasting with the worse precision score at 71%. These results indicate that ANN is overemphasizing the positive class, classifying more positive instances correctly, increasing recall, but at the cost of misclassifying negative instances, decreasing precision. This results in a significant amount of false positives, which is confirmed by ANN confusion matrix in fig. 3.

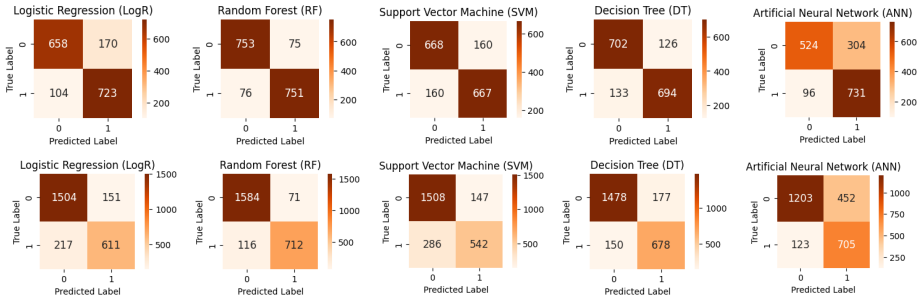

Fig. 3: Confusion matrices for each model trained with the datasets with equal (top panel) and double (bottom panel) ratio of negatives/positives. LogR, RF, SVM, DT, and ANN confusion matrices, from left to right, respectively.

As seen in table 4, DT shows slighter lower results compared to RF, with scores of approximately 85%. DT tends to over fit without proper pruning, which may justify its lower performance comparing to RF, that is not prone to overfitting [17]. DT also performs better with few relevant features and no complex interactions [17]. If features built with Propythia descriptors are not relevant

enough, it may be resulting in a drop in performance for DT. Explore other Prophytia descriptors could be an option to try to improve DT performance. As seen in table 4, LogR has similar results to DT, with scores ranging from 83% to 87%. LogR normally performs well when the features are informative and not highly correlated. If features built with Propythia descriptors are not relevant enough or have redundancy, it may explain LogR lower performance compared to RF. LogR also requires a large amount of data to have a good performance, which could be another reason for its results. SVM revealed even lower scores than the previous models, with all metrics scoring 81%, while ANN had the worst performance with all metrics bellow 80%, except recall, as mentioned previously. SVM and ANN are especially feature sensitive, requiring careful hyperparameter tuning. In addition, ANN requires significant amounts of data to train effectively [17], which may not be our case. Right hidden layer size is difficult to determine for MLP, the algorithm used for the ANN model, which is a crucial hyperparameter that would benefit a better optimisation. In fact, our models' build still has a lot of room to improve hyperparameter tuning, which would probably improve models' performance, especially for SVM and ANN. Analysing the ROC curves and respective AUC in Fig. 4, we confirm the results obtained previously, allowing for a more visual analysis of the models performance.

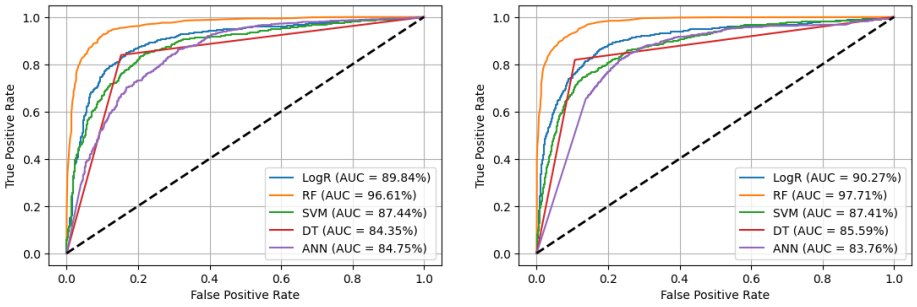

Fig. 4: ROC curves and respective AUC for each model. Models trained with the datasets with equal (left plot) and double (right plot) ratio of negatives/positives.

4 Conclusion

This work aimed at building an endolysin dataset suited for ML models training and developing them to be able to predict if phage protein sequences are endolysins or not. After building and evaluating these models, including hyperparameter tuning, we can conclude that depending on the model and the features used, different models show different performances, with the best performing algorithm being RF scoring 91% in all metrics. Even though results are satisfactory, we are still far from developing a consistent endolysin predictor

tool. In the future, this work would benefit from a more careful hyperparameter tuning, using more features or even different ones, for example using other protein descriptors, rather than only the physicochemical from Propythia. Eventually test other algorithms that may fit our data better, and try to understand if our data is suitable for our task, or if we need to perform a manual curation of the dataset. We also expect to make the model capable of predicting endolysins in phage genomes and metagenomic sequencing data, and even predict different types of endolysins, however, these are long term aims. The code developed in this project, alongside with all supplementary files can be accessed on the project's GitHub repository.

Bibliography

[1] Fatma Abdelrahman, Maheswaran Easwaran, Oluwasegun I. Daramola, Samar Ragab, Stephanie Lynch, Tolulope J. Oduselu, Fazal Mehmood Khan, Akomolafe Ayobami, Fazal Adnan, Eduard Torrents, Swapnil Sanmukh, and Ayman El-Shibiny. Phage-Encoded Endolysins. *Antibiotics*, 10(2):1–31, feb 2021.

[2] Qifang Bi, Katherine E. Goodman, Joshua Kaminsky, and Justin Lessler. What is Machine Learning? A Primer for the Epidemiologist. *American Journal of Epidemiology*, 188(12):2222–2239, dec 2019.

[3] Yves Briers. Phage Lytic Enzymes. *Viruses*, 11(2), feb 2019.

[4] Jesse Cahill and Ry Young. Phage Lysis: Multiple Genes for Multiple Barriers. *Advances in virus research*, 103:33, jan 2019.

[5] Bjorn Criel, Steff Taelman, Wim Van Criekinge, Michiel Stock, and Yves Briers. PhaLP: A database for the study of phage lytic proteins and their evolution. *Viruses*, 13(7), jul 2021.

[6] Moïra B. Dion, Frank Oechslin, and Sylvain Moineau. Phage diversity, genomics and phylogeny. *Nature reviews. Microbiology*, 18(3):125–138, mar 2020.

[7] Limin Fu, Beifang Niu, Zhengwei Zhu, Sitao Wu, and Weizhong Li. CD-HIT: accelerated for clustering the next-generation sequencing data. *Bioinformatics*, 28(23):3150, dec 2012.

[8] Hongquan Gou, Enhao Li, Yilun Xue, Yi Rong, Yihui Zhang, Cheng Chang, Wennan Guo, Shiyun Wang, Jingyang Tu, Chao Lv, Min Li, Jiewen Huang, Xiaokui Guo, Qingtian Li, and YongZhang Zhu. PhalydDB: An Extensive Phage-Derived Lytic Protein Database for Targeted Antimicrobial Engineering Design and Bacterial Host Prediction. *SSRN Electronic Journal*, jul 2022.

[9] Liliam K. Harada, Erica C. Silva, Welida F. Campos, Fernando S. Del Fiol, Marta Vila, Krystyna Dąbrowska, Victor N. Krylov, and Victor M. Balcão. Biotechnological applications of bacteriophages: State of the art. *Microbiological Research*, 212-213:38–58, jul 2018.

[10] Katarina Hojckova, Matej Stano, and Lubos Klucar. phiBIOTICS: catalogue of therapeutic enzybiotics, relevant research studies and practical applications. *BMC microbiology*, 13(1), 2013.

[11] Elliot J. Lefkowitz, Donald M. Dempsey, Robert Curtis Hendrickson, Richard J. Orton, Stuart G. Siddell, and Donald B. Smith. Virus taxonomy: the database of the International Committee on Taxonomy of Viruses (ICTV). *Nucleic Acids Research*, 46(D1):D708–D717, jan 2018.

[12] Yousef Nami, Nazila Imeni, and Bahman Panahi. Application of machine learning in bacteriophage research. *BMC microbiology*, 21(1):193, jun 2021.

[13] National Center for Biotechnology Information. Ncbi protein database. `https://www.ncbi.nlm.nih.gov/protein/?term=`, 2024.

[14] Hugo Oliveira, Luís D. R. Melo, Sílvio B. Santos, Franklin L. Nóbrega, Eugénio C. Ferreira, Nuno Cerca, Joana Azeredo, and Leon D. Kluskens. Molecular aspects and comparative genomics of bacteriophage endolysins. *Journal of virology*, 87(8):4558–4570, apr 2013.

[15] Ana Marta Sequeira, Diana Lousa, and Miguel Rocha. ProPythia: A Python package for protein classification based on machine and deep learning. *Neurocomputing*, 484:172–182, may 2022.

[16] Sonika Sharma, Soumya Chatterjee, Sibnarayan Datta, Rishika Prasad, Dharmendra Dubey, Rajesh Kumar Prasad, and Mohan G. Vairale. Bacteriophages and its applications: an overview. *Folia microbiologica*, 62(1):17–55, jan 2017.

[17] Amanpreet Singh, Narina Thakur, and Aakanksha Sharma. A review of supervised machine learning algorithms. *International Conference on Computing for Sustainable Global Development*, 2016.

[18] Dann Turner, Andrey N. Shkoporov, Cédric Lood, Andrew D. Millard, Bas E. Dutilh, Poliane Alfenas-Zerbini, Leonardo J. van Zyl, Ramy K. Aziz, Hanna M. Oksanen, Minna M. Poranen, Andrew M. Kropinski, Jakub Barylski, J. Rodney Brister, Nina Chanisvili, Rob A. Edwards, François Enault, Annika Gillis, Petar Knezevic, Mart Krupovic, Ipek Kurtböke, Alla Kushkina, Rob Lavigne, Susan Lehman, Malgorzata Lobocka, Cristina Moraru, Andrea Moreno Switt, Vera Morozova, Jesca Nakavuma, Alejandro Reyes Muñoz, Jānis Rūmnieks, Bl Sarkar, Matthew B. Sullivan, Jumpei Uchiyama, Johannes Wittmann, Tong Yigang, and Evelien M. Adriaenssens. Abolishment of morphology-based taxa and change to binomial species names: 2022 taxonomy update of the ICTV bacterial viruses subcommittee. *Archives of Virology*, 168(2):1–9, feb 2023.

[19] Maria Vieira, José Duarte, Rita Domingues, Hugo Oliveira, and Oscar Dias. PhageDPO: Phage Depolymerase Finder. *bioRxiv*, page 2023.02.24.529883, feb 2023.

[20] Mauno Vihinen, Esa Torkkila, and Pentti Riikonen. Accuracy of protein flexibility predictions. *Proteins*, 19(2):141–149, 1994.

[21] Peter J. Walker, Stuart G. Siddell, Elliot J. Lefkowitz, Arcady R. Mushegian, Evelien M. Adriaenssens, Poliane Alfenas-Zerbini, Donald M. Dempsey, Bas E. Dutilh, María Laura García, R. Curtis Hendrickson, Sandra Junglen, Mart Krupovic, Jens H. Kuhn, Amy J. Lambert, Małgorzata Łobocka, Hanna M. Oksanen, Richard J. Orton, David L. Robertson, Luisa Rubino, Sead Sabanadzovic, Peter Simmonds, Donald B. Smith, Nobuhiro Suzuki, Koenraad Van Doorslaer, Anne Mieke Vandamme, Arvind Varsani, and Francisco Murilo Zerbini. Recent changes to virus taxonomy ratified by the International Committee on Taxonomy of Viruses (2022). *Archives of Virology*, 167(11):2429–2440, nov 2022.

[22] Hongyu Wu, Jinjiang Huang, Hairong Lu, Guodong Li, and Qingshan Huang. GMEnzy: A Genetically Modified Enzybiotic Database. *PLoS ONE*, 9(8), aug 2014.

[23] Hongyu Wu, Hairong Lu, Jinjiang Huang, Guodong Li, and Qingshan Huang. EnzyBase: a novel database for enzybiotic studies. *BMC microbiology*, 12(1):54, dec 2012.