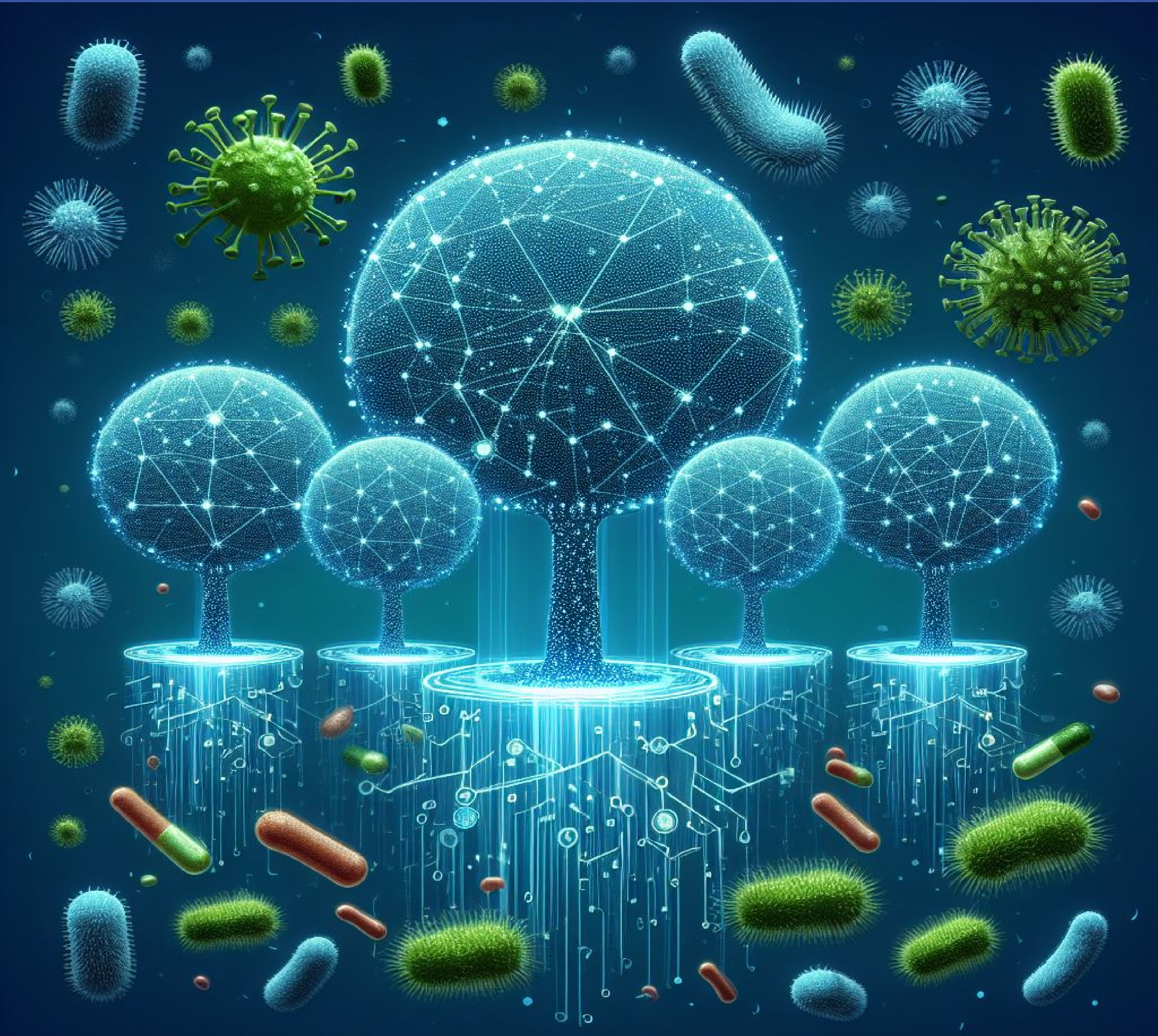
DTU Bioinformatics and Systems Biology
Department of Health Technology

Predicting Phage-Host Interaction with Language Models

Master Thesis

Paolo Federico - s212975



Predicting Phage-Host Interaction
with Language Models

Master Thesis
January, 2024

By
Paolo Federico - s212975

Published by: DTU, Department of Health Technology, Ørsteds Pl., Building 345, 2800 Kgs. Lyngby Denmark
https://www.healthtech.dtu.dk/

Approval

This thesis has been prepared over five months at the Department of health Technology, at the Technical University of Denmark, DTU, in collaboration with the Novo Nordisk Center For Protein Research, at the Copenhagen University, KU, in partial fulfillment of the degree Master of Science in Engineering, MSc Eng.

It is assumed that the reader has a basic knowledge in the areas of machine learning and biology.

Paolo Federico - s212975 - s212975

[illegible]

Signature

28 January 2024

Date

Abstract

The escalating challenge of antimicrobial resistance has prompted increased interest in utilizing phages to combat bacterial infections. This paper addresses the prediction of phage-bacterium protein-protein interactions, specifically focusing on the binary classification of interactions between a phage receptor-binding protein and the receptor protein of Escherichia coli K12. In contrast to prior approaches requiring manual feature engineering, the presented method relies solely on protein sequences.

The interaction prediction is formulated as a binary classification problem, using the protein sequences of the phage receptor-binding protein and the bacterium receptor protein as input. Two large language models, ProtT5 and ProtXLNet, are explored to encode these sequences automatically. These models generate dense embeddings without additional alignment or structural information. The classification is executed using random forests.

The performance of the proposed method is compared with previous works where prediction focused on host genus or species based on viral protein sequences. This comparative analysis sheds light on the effectiveness of this approach in the broader context of phage-bacterium interactions and contributes to the ongoing efforts to address antibiotic resistance.

All data and programs used in the projects are available at
https://github.com/Drivahah/Thesis [1].

Acknowledgements

Paolo Federico - s212975, MSc Bioinformatics and Systems Biology, DTU

Henrik Nielsen, Assoc. Prof., Supervisor

Nicholas Taylor, Assoc. Prof., Co-supervisor

Simon Rasmussen, Assoc. Prof., Co-supervisor

Felix Teufel, PhD, Co-supervisor

Contents

Preface ii
Acknowledgements iv

1 Introduction **1**
1.1 History of Pandemics 1
1.2 Resistance 2
1.3 Novel Strategies 3
1.4 Infection 4
1.5 Protein Interactions 5
1.6 Machine Learning 6

2 Materials and Methods **7**
2.1 Python 7
2.2 Datasets 8
2.3 Oversampling techniques 9
2.4 Protein Language Models 11
2.5 Classifier 13
2.6 Nested cross-validation and grid-search 14
2.7 Procedure 15

3 Results and Discussion **17**
3.1 Dataset composition 17
3.2 Pipelines compared 17
3.3 About thresholds 19
3.4 Recall the hand-crafted features 24
3.5 It takes confidence 26
3.6 Biological matters 27
3.7 Space for improvement 28

4 Concluding remarks **31**

Bibliography **33**

A Appendix **37**
A.1 Probability distributions of each pipeline 37
A.2 Scoring and fitting times for each pipeline 39

1 Introduction

This thesis aims to predict interactions between a phage receptor-binding protein and a bacterial receptor protein, addressing the challenges of infectious diseases and the escalating threat of antibiotic resistance. The focus is on exploring innovative solutions, specifically employing language models and a random forest classifier.

The introduction dives into the historical context of infectious diseases, emphasizing the potential of viral therapies as a response to antibiotic resistance. It provides a concise overview of bacteriophages, bacteria, and infection mechanisms, followed by a brief exploration of proteins and their interactions.

In the last few decades, research methodologies shifted toward statistical analysis. The primary objective of this thesis is to employ advanced computational techniques to predict interactions between phage receptor-binding proteins and bacterial receptor proteins. The integration of language models and a random forest classifier aims to bridge the gap between traditional wet lab approaches and the evolving field of computational biology and provides new tools for the understanding of infection mechanisms.

1.1 History of Pandemics

Humans, throughout history, have been fighting diseases for their survival. The first recorded pandemic is the plague of Justinian, dated 541-542 AD, which spread from Egypt to Constantinople and claimed the lives of 30-50 million people [2]. In the following centuries, epidemics were quite rare, until the outbreak of the Black Death. Between 1348 and 1350 it killed at least a quarter of Europe's population, winning the record for the worst disaster that has ever fallen on humankind. Eighty percent of those who contracted the disease died in a couple of agonizing days. In England the pandemic reached its peak with the Bubonic Plague of 1665, decimating London's population fig. 1.1. The origin of the sickness was unknown. God's punishment for human sins, that was the common belief. Some people depicted it as a retribution delivered through the poisoned arrows of evil angels, “venomous moleculae” or comet-borne miasmas [3].

After all, a less metaphoric and picturesque explanation would have been impossible before the invention of the microscope, in the 17th century. Two pioneers adventured in the world of microscopic creatures: Robert Hooke and Antoni van Leeuvenhoek. In 1665 Hooke published Micrographia, where he describes microorganisms and proposes the first depiction of one of them: the microfungus Mucor fig. 1.2. Van Leeuvenhoek in 1676 was wondering about the taste of pepper. After analyzing some “pepper water” that had been sitting for three weeks, he found some organisms of 1-2 micrometers in size, bacteria. Or, “animalcules”, how he named them back then [4].

Not long before that, in 1658, Athanaseus Kircher had published his *Scrutinium pestis physico-medicum*, inventing the germ theory of disease [5]. However, the born of medical bacteriology had to wait for Louis Pasteur and Robert Koch. In the 1860's the first was conducting his studies on fermentation and pasteurization, while the latter developed techniques to grow and study bacteria that are still in use nowadays [6] [7]. In those years another disease made its appearance: cholera. The first European pandemic took place in 1830-1837 and was first originated in the Ganges delta in 1817. It was Koch himself who detected the bacterium *Vibrio cholerae* and identified it as the cause of the disease, in 1883. In the first years, Denmark was initially kept safe from the disease placing military

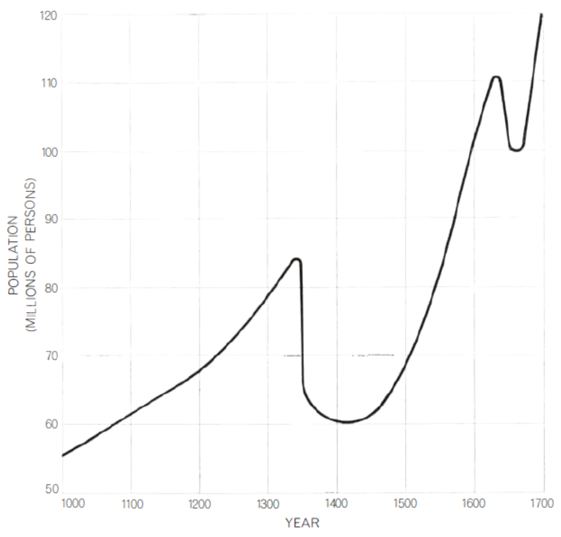


Figure 1.1: Europe population through time. The impact of plagues in 1348 and 1665 is obvious [3].

barriers at the border with Germany and using the Saltholm island as a quarantine station for those who planned to Zealand. The year after the abolition of quarantine, in 1853, the pandemic reached Copenhagen and was not over for 100 years. At the time, the city had latrine pits that were emptied twice a year, and the construction of efficient sewers took 40 years. On top of that, the drinking water came from the uncleaned surface of the lakes and was transported in wood gutters to posts on the streets. *Vibrio cholerae* found in stagnating water a perfect habitat, infecting the people who drank it. The Cholera epidemic still happens in the present day, especially when natural disasters, wars, or other factors force poor hygienic conditions, causing thousands of deaths [8].

We now know that some of the biggest pandemics throughout history were caused by bacteria, from *Yersinia pestis* for the Justinian plague [2] to the *Bacillus pestis*, which caused the Black Death, [3] and to *Vibrio cholerae*.

The first remedies against infections have been known since ancient times. Mushrooms, beer yeast, and molds were deemed valuable for treatments at least 3500 years ago. After the discovery of microorganisms, many people noticed that molds would limit the growth of neighboring bacteria. The first successes in obtaining and using antibiotics from microorganisms took place at the end of the 19th century. Some Arabian hostlers were storing their saddles in dark and humid rooms. Mold grew on the seats and that seemed to heal sores. Ernest Duchesne, a French medical officer, managed to cure *Escherichia coli* infected guinea pigs with a suspension of the aforementioned mold. However, the large-scale production of the antibiotic obtained from the mold *Penicillum chrysogenum*, known as penicillin, happened only in 1940 thanks to the work of Alexander Fleming, Howard Walter Florey, Ernst Boris Chain, and Norman Heatley [9]. In 30 years, the life expectancy in the US extended from 47 to more than 70 years [10].

1.2 Resistance

Even though in the new millennia medical treatments and prevention have exponentially improved, future bacterial pandemics are not to be excluded. Some conditions may enhance the risk of spread: a growing population living in dense megacities, fast traveling and migration increase the number of interactions between people. Humans in close con-

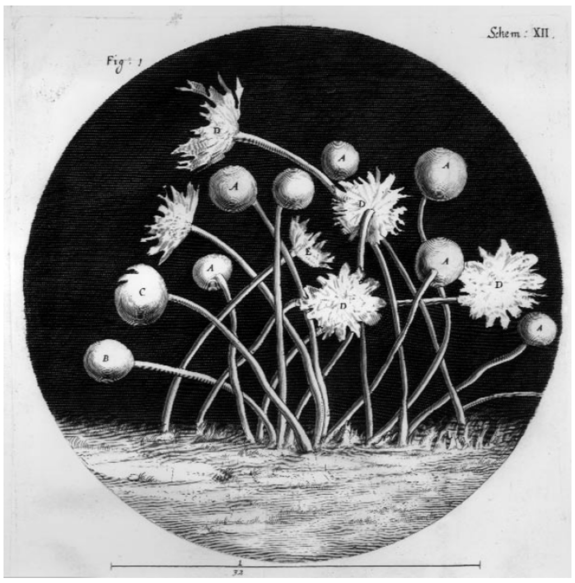


Figure 1.2: Microscopic view of a ‘hairy mold’ colony described by Robert Hooke in 1665 (in Micrographia). This image was the first published depiction of a microorganism [4].

tact with many animals can offer the opportunity for animal bacteria to adapt and cross the species, and climate change can lead to the migration of infectious microorganisms and their hosts. Finally, wars and natural disasters can destroy the infrastructures necessary for hygienic conditions [8].

And it’s not the end of the story. In 1929 Fleming had already realized that some microbes were not affected by penicillin. A few years later, Chain isolated an *E. coli* product that destroyed the growth-inhibiting property of the drug [9]. The medical application of antibiotics saved uncountable lives in the past hundred years, but on the other hand, it acts as an unprecedented selective pressure on bacteria. The result is an increase in the number of organisms that are immune to multiple drugs. Bacterial evolution can take place in two ways: genome mutations and horizontal transfer. In one case, a spontaneous mutation in the DNA can result in the acquisition of new functions, while in the other the new functions are acquired from other organisms in the surrounding environment. Broad-spectrum antibiotics are abundantly used, even for treating generic infections, which means that the pathogen organisms are the ones developing the most resistance genes. Multidrug-resistant infections are difficult or impossible to treat, and the number of new treatments is limited [11], [12]. On top of that, higher ambient temperatures seem to correlate with a faster spread of resistant pathogenic bacteria [13].

1.3 Novel Strategies

Due to the ongoing battle with microorganisms, the world is seeking novel strategies. Some attempts focus on blocking the efflux pump of the bacterium, prohibiting it from getting rid of the antibiotic [14]. Targeting the pump used for taking the lipopolysaccharide - a fundamental constituent of the cell membrane - to the surface can affect the vitality of the organism [15]. Among the most promising directions, we find antimicrobial oligonucleotides, monoclonal antibodies, and phage therapy.

The oligonucleotides are short nucleic acid sequences that can pair with other DNA strings and can be used to silence deleterious genes. They could be employed with any kind of pathogen, and do not suffer from antibiotic resistance mechanisms. The sequence can easily be updated in case of mutation of the target. The big size of the molecule prolongs its time inside the cell but might make it more susceptible to mechanisms of resistance that would avoid its entrance.

The monoclonal antibodies are large complex molecules that act externally, binding to the antigen - a molecule specific to the pathogen agent - and neutralizing its function. The infection is subsequently cleared by the immune system. Resistance to antibodies usually involves some secreted proteases or membrane-bound proteins. Both the oligonucleotides and the antibodies have the potential to be obtained with a much faster and more efficient drug discovery process, although initial investments and technology development would be needed [16].

Phages are viruses that exclusively infect bacteria, and exploiting the bactericidal action of these viruses offers a therapeutic approach. Often, phages have very high specificity for one host, to the point that the same virus could be effective on one patient, but not on another presenting the same infection. It is necessary thus to either select viruses with a broad host spectrum or brew phage cocktails that would contain a mix of phages with different targets. Phages and bacteria are natural antagonists, and as such, the latter can evolve defense mechanisms. Anyway, it is unlikely for a microorganism to employ many different resistance apparatus at the same time. Once again, a phage cocktail would enable to attack the cell on multiple fronts. Alternatively, viruses can be administered sequentially. It must be noted that the patient's immune system could learn to neutralize the virus, reducing its efficacy. A sequential pattern, rather than a simultaneous mix, would thus extend the potential treatment regimen. Viruses can show complex behaviors and particular care must be taken. First of all, we should make sure that its genome does not encode any product that would be toxic for the patient. Not only that, it should also not be a generalized transducing phage, meaning that the phage should not enable processes for the transfer of resistance genes. There are some advantages to adopting a phage treatment: First, the viruses could interact - directly or indirectly - with the antibiotics. Even though both synergism and antagonism are theoretically possible, the last case seems rare. Conversely, in several instances, the phages lowered the minimum inhibitory concentration of antibiotics for drug-resistant strains. Secondly, viruses are an attractive alternative for cases where other molecules would be of little use. For example, the biofilms. Some bacteria can grow in complex colonies - also composed of different strains - where various cells play different functions to form a structure called biofilm. While drug molecules struggle to pass through this formation, viruses can produce enzymes suited for the film disruption. Lastly, viruses may be used as vectors to deliver nucleases, enzymes engineered to target antibiotic resistance, or virulence genes, eventually causing the death of the bacterial cell. Viruses can be found in abundance and with an enormous biodiversity in nature, and their isolation is rather easy. Also, the production upscale of purified phages is relatively easy and inexpensive [10], [16].

1.4 Infection

Bacteriophages, often referred to as "phages," are natural predators of bacteria that rely on the host's replication apparatus. In simple terms, the infection cycle follows this order: adsorption, phage genome insertion and replication, viral particle production, and host cell lysis. The key to the success of the infection process lies in phage adsorption, a critical step initiated by interactions between phage proteins and bacterial surface receptors. Through these interactions, the phage recognizes the host and gets in place for the following DNA injection. The adsorption is fundamental in determining host range specificity. This mechanism has a big level of complexity. The receptors are of various kinds: proteins, sugars, and cell surface structures. From peptide sequences to polysaccharide molecules, the nature and location of these receptors vary widely depending on both the phage and its host. A main role is played by receptor-binding proteins (RBPs) located at the tails of bacteriophages. These RBPs, which can be named tail spikes, tail fibers, or spike proteins depending on the phage type, are central to deciding phage specificity. Due to these specific interactions, most bacteriophages exhibit a limited host range, targeting a particular species or strain without affecting the broader microbial community. This characteristic makes them good candidates to target pathogenic bacteria in a therapeutic context [17].

However, detailed information about the adsorption mechanism is hard to gain. Traditional techniques, such as plaque assays, struggle to examine individual interactions between specific phage machinery and targeted bacterial receptors. These methods often rely on the use of complete phages and the visualization of cell death. The results are thus more descriptive of the effectiveness of the whole process, rather than of the initial interaction [18].

Protein-protein interaction is not the only mechanism for cell adhesion. Often, the virus interacts with the lipopolysaccharide (LPS), one of the main components of the cell membrane. Not only, another frequent receptor is the O-antigen. Anyway, the focus of this thesis stays on the protein receptors. Interestingly enough, the viral genome composition varies in function of the species they infect. This characteristic enables the association with bacteria in the evolutionary tree. On top of that, the DNA similarity between the host and phage seems to be involved in the success of infection [19].

1.5 Protein Interactions

Protein-protein interactions (PPIs) are intricate processes that happen in the intricate cellular context. They involve various interaction partners such as other proteins, nucleic acids, small molecules, peptides, carbohydrates, and fatty acid chains. Not only do proteins form connections in this environment, but they also keep a high degree of specificity. At the core of understanding PPIs lies the interface between interacting proteins. Most of it is engaged in direct contacts, and the rest is occupied by water molecules. In other words, there is a high degree of shape complementarity. Also, amino acid chemistry must be suitable to promote the connection. However, the computation of possible interactions remains challenging. Indeed, various factors add layers of complexity. First, the competing ligands. The bound complex, as compared to the detached forms, can be described as the two sides of a chemical reaction. The complex concentration at the equilibrium defines the affinity of the molecules. In the complex cellular environment, protein concentrations and competing interactions add a layer of complexity. For instance, a weaker interaction with a higher concentration may outcompete a potentially stronger interaction, affecting the formation of protein complexes. Secondly, post-translational modifications and co-factors further complicate the PPI landscape, potentially switching a protein from a binding to a non-binding state. The protein association process involves the formation of an initial complex stabilized by long-range electrostatic interactions. This initial complex can lead to a more stable connection, which includes desolvation and the formation of short-range interactions.

Some observations on the aminoacidic composition can give hints about the final protein structure and dynamics Permanent PPIs tend to bury a larger surface area and exhibit more hydrophobicity compared to transient PPIs. Interface residues are evolutionarily more conserved than the others. This offers an idea of how crucial is their role in stabilizing the complex. The question of whether the detailed architecture of a binding site is imprinted in the unbound state of a protein is explored through the concept of hot spot residues. These residues, often pre-organized in the unbound state, contribute to the theory of protein–protein binding supersites, suggesting the presence of generic binding hot areas. This insight into the structural and dynamic aspects of PPIs contributes to the ongoing efforts to comprehend and predict these interactions in the context of RBP-RP interaction [20].

1.6 Machine Learning

The term ”machine learning” was introduced by Arthur Samuel, who conceived it as granting computers the ability to learn without explicit programming. This approach provides some algorithms able to tune a model's parameters depending on the available data, whose quality is fundamental to achieving a successful model. The process of acquiring knowledge from data is called ”training”. There are two principal types of machine learning: supervised and unsupervised. Supervised learning aims to predict an outcome or target, with a focus either on classification or regression. Unsupervised learning, on the other hand, deals with unclassified features. Its use is the identification of relationships or patterns within the data [21].

In the medical realm, machine learning is increasingly applied to address various challenges, leveraging large digital health record datasets. This technology proves particularly promising in predicting and combating antibiotic resistance and tailoring treatments with the abundance of genome sequence data harvested thanks to next-generation sequencing techniques. In clinical virology and microbiology, traditional methods like culturing and polymerase chain reaction (PCR) have long been fundamental and useful in defining infections and treatments. However, they are limited in speed, scalability, and adaptability. Artificial Intelligence (AI) offers rapid, real-time analysis through machine learning algorithms. This not only accelerates diagnostics but also has implications for treatment personalization, predicting responses to antimicrobial drugs, and optimizing treatments.

Deep Learning (DL), a specialized subset of machine learning, excels in handling complex medical data. The field is the current promise in refining and speeding up diagnostics, reducing the chances of false negatives or positives. AI has the potential to discover complex patterns in viral and cellular genomes and proteomes, deepening our understanding of infections and their prevention. Beyond diagnostics, AI holds considerable potential in precision medicine, where medical treatments are customized to individual patient needs based on the intricate interplay of genomics and disease manifestation. Furthermore, AI's robustness in data mining and pattern recognition is revolutionizing drug discovery, enabling rapid screening of compound libraries and predicting drug efficacies [22].

2 Materials and Methods

2.1 Python

The whole analytical process was conducted using Python, a popular high-level programming language. One of its main advantages is the extensive availability of packages that let users perform complex tasks. Among the ones used in the project, some are worth citing:

- NumPy is an array programming tool that provides powerful manipulation of data in vectors, matrices, and higher-dimensional arrays. It is the foundation upon which the scientific Python ecosystem is constructed [23]
- Pandas library implements working with structured datasets, it provides routines for performing common data manipulations and analysis. Several observations constitute a dataset, which is usually constructed so that each sample occupies a row, and their descriptive features, or attributes are stored in separate columns. Pandas' basic data structure is the dataframe, an indexed dataset in which every column is a NumPy array object [24]
- Scikit-learn integrates many machine learning algorithms for medium-scale supervised and unsupervised tasks. The core object is the estimator. It always implements a "fit" method to train the model and, in the supervised scenario, a "predict" method. Some estimators - called transformers - embody a transform method, returning modified input data. The score method returns an estimation of the model's goodness. The package also provides a cross-validation iterator (see cross-validation paragraph) and backs up model selection with the "GridSearchCV" object, that scans through all the possible combinations of the provided model parameters. Lastly, the "Pipeline" object combines several transformers and a final estimator and can be treated as a sole estimator [25]
- Imbalanced-learn is fully compatible with scikit-learn and copes with the problem of imbalanced datasets (see oversampling techniques paragraph). The main object is the "sampler", and its "fit_sample" method first calculates some statistics which are then evaluated to resample the data, which is returned with a balanced ratio [26]

The code to run the project's analysis was written with the aid of Microsoft Visual Studio Code, a highly versatile and user-friendly code editor. It possesses a strong ecosystem of extensions. One of these enables the IPython notebooks, that provide support for data visualization and parallel computation [27]. The notebooks are organized in cells containing either code, Markdown text, or visualizations. Such a structure is ideal for integrating code with documentation. The data wrangling part of the project was carried out in IPython notebooks.

Given the nature of the project, an important computation capacity needs to be utilized. The High-Performance Computing services provided by DTU's Computing Center were employed. The nodes are accessible with a DTU account, via a Unix interface. To run some code, it must be submitted to the queueing system using a bash script where some options are specified [28]. The heavy statistical models on which the project relies can be hastened thanks to parallel computation run on a graphics processing unit (GPU). Specifically, a Tesla V100 32GB was exploited [29].

2.2 Datasets

2.2.1 BASEL receptors

The project makes extensive use of the data provided by Maffei et al. [30]. In the study, a systematic exploration of the phages infecting *E. coli K-12 MG1655 ΔRM* was conducted. More specifically, the strain K12 of the bacterium. They gathered data about 68 of such viruses in the BASEL collection (BActeriophage SElection for your laboratory). Additionally, this library includes 10 well-studied traditional model phages. The “BASEL receptors” dataset includes all 78 viruses and associates each of them with their known host-receptors. It is defined by 7 features. Each virus is described with an identifier (”bas”), and the attributes “genus”, “phage”, “morphotype”, and “closest relative”. For the BASEL viruses, the identifier is “BasXX”, where XX is a number. The model phages didn’t initially have an identifier, so for the sake of the analysis, it was manually derived from the name found in the “phage” column. The remaining three features describe the virus’s interactions with the host, namely the “primary receptor” and “terminal receptor”. A sample of the dataset is found in table 2.1.

bas	genus	phage	morphotype	closest relative	primary receptor	terminal receptor
Bas01	Rtpvirus	Escherichia phage AugustePiccard	siphovirus	RTP (AM156909.1)	LPS / O-antigen?	LptD
Bas02	Guelphvirus	Escherichia phage JeanPiccard	siphovirus	CEB_EC3a (NC_047812.1)	LPS / O-antigen?	LptD
Bas03	Guelphvirus	Escherichia phage JulesPiccard	siphovirus	CEB_EC3a (NC_047812.1)	LPS / O-antigen?	FhuA
Bas04	Warwickvirus	Escherichia phage FritzSarasin	siphovirus	tonnikala (NC_049817.1)	LPS / O-antigen?	BtuB

Table 2.1: BASEL receptors

2.2.2 BASEL proteome

The BASEL proteome made available from [30] includes the whole sets of protein sequences for each phage included in the previous dataset. The file comes in FASTA format, containing a list of aminoacidic strings. Each sequence is defined by its header, which states the available information about it. It looks like this:

```
>lcl|MZ501051.1_prot_QXV76132.1_1 [locus_tag=bas01_0001] [protein=
    terminase small subunit] [protein_id=QXV76132.1] [location=1..507]
    [gbkey=CDS]
MSKAALKMGEGNFKALYNKKYGDIAMVAINRKYTPEEVFDFAVRYFSWAESEAIKAIETAAFQGVVSESL
VHKPRVFTLTGLALFMGVNINRFARWRTEAGYSDVMEFIDNVIYEQKYQLGVAGIINSTIVGKELGIDKP
QEITISNNSTANDVDSMKEALESVISKL
>lcl|MZ501051.1_prot_QXV76133.1_2 [locus_tag=bas01_0002] [protein=
    hypothetical protein] [protein_id=QXV76133.1] [location=526..621] [
    gbkey=CDS]
MKGFIKLFIWYYLLTSISLCVFMLVVKLWLI
>lcl|MZ501051.1_prot_QXV76134.1_3 [locus_tag=bas01_0003] [protein=
    terminase large subunit] [protein_id=QXV76134.1] [location
    =609..2177] [gbkey=CDS]
```

```
MANLIWEEMTSQEKLAVKAISEHSFEGFLRCWFSITQGERYIPNWHHKYLCRIIDEIIAGERKDTIINIA
PGSGKTEIASIHFPAYSMVKLKKVRNLNISFADSLVKRNSKRVRDLIKSQEFQSLWPCKFGTCKDDELQV
LDENGRVRFESISKAAGGQITGARGGYITETYSGAVLLDDFDKPADMLSQVVYRTNNHVMLKNTIRSRRAS
SVKGKATPIISIQQRLHVNDSTWFMMEGGMGIDFDLIKIPALVTEDYIDSLPDWIKEQFADDVLSSEYIE
RDGVKYYSYFPKKESVNDLVAMWDSDPYTFLSQYQQEPVALGGNLINVDWFQRISDTFRPPAKYDYRFIT
CDTAMTTKSYSDFSVLQLWGYKDAKIYLIDQRRGKWEAPELESELLDFERKSRATSQTDGILRKIIIEKK
ASGIGLIQSAGRVMRTPIEPYVPDNDKLTRIMSALPQIKAGNVVLPESAPWLNGLLTEVAAFTADDSHKH
DDQIDCLTMAVNLVLNIADDPKSRMLRLAGIK
```

Listing 2.1: BASEL proteome FASTA file

Every header starts with “>” and a unique identifier, followed by tags between square brackets. The sequence begins in the new line and is split into 70 characters-long strings. A user-defined python function allowed to collect this information in a dataframe, with the identifier, the sequence, and the various tags in separate columns. Most importantly, the “locus tag” associates each protein with a virus in the format “[virus id]_[protein number]”. A “bas” identifier column was later derived from “locus_tag” to be able to associate each protein with the respective phage in the BASEL receptors dataset. Another important attribute obtained from the FASTA file is “protein”, which gives a short description of the protein’s role. table 2.2 includes a sample of the dataframe.

seqID_phage	locus_tag	protein	protein_id	location	gbkey	sequence
lcl|MZ501051.1_prot_QXV76132.1_1	bas01_0001	terminase smallsubunit	QXV76132.1	1..507	CDS	MSKA...
lcl|MZ501051.1_prot_QXV76133.1_2	bas01_0002	hypotheticalprotein	QXV76133.1	526..621	CDS	MKGF...
lcl|MZ501051.1_prot_QXV76134.1_3	bas01_0003	terminaselargesubunit	QXV76134.1	609..2177	CDS	MANL...
lcl|MZ501051.1_prot_QXV76135.1_4	bas01_0004	putativehomingendonuclease	QXV76135.1	2321..2716	CDS	MVAG...
lcl|MZ501051.1_prot_QXV76136.1_5	bas01_0005	hypotheticalprotein	QXV76136.1	2788..3168	CDS	MTKK...

Table 2.2: BASEL proteome dataframe

2.2.3 K12 proteome

The proteome of *E. coli K-12 MG1655 ΔRM* could be found in UniProt as a FASTA file, at [31]. UniProt is a knowledgebase with the aim to provide a comprehensive, high-quality, and freely accessible set of protein sequences annotated with functional information [32]. The previously stated Python method was also adopted for the creation of a dataframe for the bacterial proteome. The “GN” column contains the gene identifier for each protein and will link the protein sequence to the receptors stated in the BASEL receptors dataset. See table 2.3

2.3 Oversampling techniques

Many bioinformatic datasets present a binary classification. Some examples might be the identification of a tissue as cancerous or not, the responsivity of a cell line to a drug, or - like in the current scenario - the interaction among two proteins. Such a configuration gives

seqID_k12	name	OS	OX	GN	sequence
sp|A5A616|MGTS_ECOLI	Smallprotein MgtS	Escherichiacoli(strainK12)	83333	mgtS	MLGN...
sp|O32583|THIS_ECOLI	Sulfurcarrier proteinThiS	Escherichiacoli(strainK12)	83333	thiS	MQIL...
sp|P00350|6PGD_ECOLI	”6-phosphogluconatedehydrogenase,decarboxylating”	Escherichiacoli(strainK12)	83333	gnd	MSKQ...
sp|P00363|FRDA_ECOLI	Fumaratereductaseflavoproteinsubunit	Escherichiacoli(strainK12)	83333	frdA	MQTF...
sp|P00370|DHE4_ECOLI	NADP-specificglutamatedehydrogenase	Escherichiacoli(strainK12)	83333	gdhA	MDQT...

Table 2.3: *E. coli K-12 MG1655 ΔRM* proteome

origin to the “class imbalance” problem [26]. In other words, a machine learning algorithm wouldn’t succeed in completing proper training on the minority category. A classifier that only predicts the majority group with no regard for the input would still achieve a deceivingly high accuracy. Various oversampling techniques aim to increase the number of less-represented classes to produce a dataset of balanced proportions. The simple replication of existing samples usually increases the model’s performance. However, training the algorithm on identical samples easily results in overfitting, likely leading to an increase of false positives - not a desirable thing when dealing with, for example, disease diagnosis. Some more advanced techniques have been developed. The ones employed in the project are SMOTE and ADASYN [33]

2.3.1 Synthetic Minority Oversampling Technique (SMOTE)

The SMOTE algorithm proved itself robust when applied to different types of problems. Instead of proposing existing observations, it introduces synthetic instances. The vector of attributes defining a sample identifies its position in the feature space, and the key to SMOTE’s success is the addition of neighboring points to the minority class (see fig. 2.1). The algorithm follows these steps:

- An instance x_i of the minority class is selected.
- Among the same group, a number K of the nearest neighbors is identified.
- The differences between x_i and each neighbor are calculated and multiplied by a fraction between 0 and 1.
- These values are added to x_i to obtain the new samples [34]

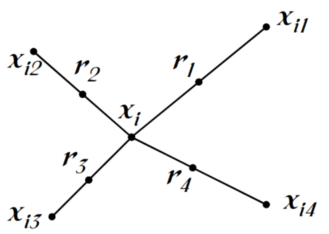


Figure 2.1: Visualization of SMOTE algorithm. Xi is the chosen observation, while Xi1-4 are the identified neighbors. r1-4 are the synthesized samples. [33]

2.3.2 Adaptive Synthetic Sampling (ADASYN)

Similarly to SMOTE, ADASYN is a synthetic data generation algorithm. On top of reducing the bias due to the class imbalance, it aims to give improved representation to the samples that would be harder to learn. Specifically, such instances are the ones located in regions of the feature space more crowded with elements of the majority class, rather than a homogeneous population of minority elements. The procedure is similar to SMOTE, but the number of synthetic points created from each sample is proportional to the major class neighbors of that sample. Roughly, these are the steps:

- For each minority sample, the ratio r_i of the majority class samples over all the neighbors is measured.
- Each r_i is normalized over the sum of all the r_i.
- For each sample, obtain the number g_i of points to generate as the normalized r_i times the total number of synthetic points necessary to balance the dataset.
- For each sample, generate g_i synthetic points from its minority neighbors as in SMOTE [35]

2.4 Protein Language Models

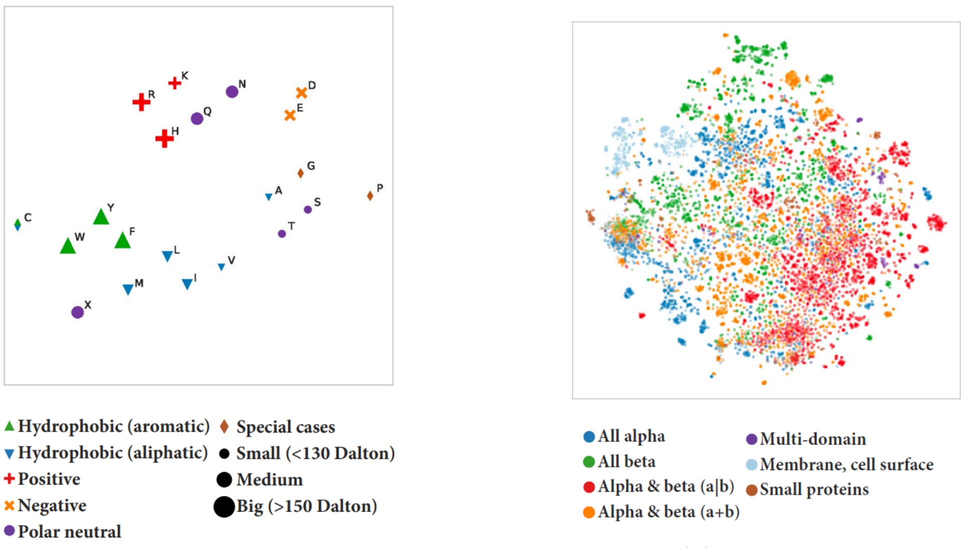
Aminoacidic sequences were converted to vectors using language models previously trained on protein data. Two different algorithms were used: ProtT5 and ProtXLNet. Many language models are constrained by the input length. An aminoacidic sequence can be thousands of residues long, and the project's input data can be as long as two concatenated proteins. ProtT5 and ProtXLNet were chosen because they don't 'present this limitation. On top of that, ProtT5 performed greatly on various tasks. The two models were imported from the Hugging Face transformers python package [36].

2.4.1 ProtT5

ProtT5 is a protein language model based on the T5 transformer architecture. T5 performs language translation using first an encoder to project sentences to vectors in an embedding space, then a decoder to convert these numerical elements to a target language. Usually, transformers require input sequences of up to a fixed length. Indeed, the self-attention mechanism they are built upon relies on positional encoding. Conversely, T5 learns a positional encoding for each attention head, enabling the model to make predictions beyond its length. [37]

Different variants of protT5 have been constructed. The one exploited here has a size of 3 billion parameters (T5-XL) and was trained on two datasets: Big Fantastic Database (BFD) and Uniref50 (UniProt Reference Cluster). BFD is the biggest collection of protein sequences, about 8 times larger than the broadest sets used previously for language tasks. UniRef50 provides clustered sets of sequences - and isoforms - from the UniProt Knowledgebase [38]. Self-supervised training was executed for ProtT5 on BFD for 1.2 million steps (ProtT5-XL-BFD), and successively on UniRef50 for 991.000 steps (ProtT5-XL-U50). ProtT5 was trained to corrupt and reconstruct single tokens of the input sequence.

The model's embeddings are produced at a per-residue level - in other words, each amino acid defines a vector. However, average pooling over the sequence dimension produces a per-protein representation of fixed-length independent from one of the sequences. Concerning the per-residue aspect, ProtT5 proved itself good in capturing amino acids' characteristics. fig. 2.2a depicts a bidimensional projection of the amino acids' encodings. Noticeably, the molecules with similar chemical features cluster together. The projections of per-protein encodings - fig. 2.2b - are also quite distinguishable among the various groups.



(a) Per-residue

(b) Per-protein

Figure 2.2: Left: Bidimensional t-SNE projection of ProtT5 amino acids encodings. The molecules with similar chemical features cluster together. Right: Bidimensional projection of ProtT5 per-protein encodings. The various groups of proteins occupy quite distinguished regions of space. [39]

ProtT5 has been tested on supervised training tasks both on per-residue and per-protein levels. This has been done using the embeddings as input for other models. State-of-the-art techniques for the prediction of the proteins' secondary structure as well as protein classification rely on multiple sequence alignment and evolutionary information. However, ProtT5-XL-U50 equaled and outperformed - respectively - such performances [39]. This model was also included in works that outpaced state-of-the-art techniques regarding phage-host interaction prediction [40]

2.4.2 ProtXLNet

ProtXLNet is built upon the XLNet decoder, without altering its original configuration, except for an increased number of layers. Oppositely to other language models, XLNet can process sequences of arbitrary length. On top of that, it can gather bidirectional context. While in the canonical natural language field uni-directional and bi-directional perform similarly, when dealing with the protein language the second configuration works best. The protein language model was trained on the UniRef100 database for 867k steps in total [39].

2.5 Classifier

In the project, the protein sequences - in a vectorized format - were input into a Random Forest classifier, described in the following paragraph

2.5.1 Random Forest

The Random Forest algorithm was conceived by Breiman in 2001 [41]. It is fundamentally an expansion of the Classification and Regression Tree (CART) technique. A tree splits the samples into smaller and smaller subgroups, aiming for groups where most - or possibly, all - the observations belong to the same class. The initial training dataset is found on the root node. Every splitting point is a branch, and the final nodes - where the algorithm stops - are the leaves. At each splitting point, a number of random features is chosen - often the square root of their count. The selected attributes are used to set some threshold values according to which divide the observations in the following nodes. All the possible thresholds are tried out, picking the ones that improve leaf purity. Depending on the task a different metric is adopted. For classification, the most popular score is the Gini impurity index. The Gini value of a leaf is calculated as the sum of the squared probabilities of each class. For example, if half of the samples belong to one class, and the other half to another class: Gini $= (0.5)^2 + (0.5)^2$ A lower score reflects a more pure group. The tree's Gini impurity index is the weighted average of all of its leaves [42].

The splitting parameters and threshold selection introduce a bias in the model. Random Forest aims to reduce the bias by pooling the predictions of many decision trees. In a classification task, the final forecast will be the majority class, i.e. the one who received the most votes from all the trees. The key to this algorithm's success is randomization and it is introduced in two levels. The first is the selection of different splitting parameters in each tree, and the second is bagging. Bagging stands for Bootstrap Aggregating and consists of a random sampling of the dataset with replacing. This sampling is performed for each tree, meaning that every single classifier will be trained on a different dataset, on which some samples might appear more than once, while others will not be seen. Every time, the process leaves out some samples, named the "out-of-bag" set. It will constitute the validation set for each tree [43].

Random Forests seem to be able to deal with small sample sizes with high-dimensional feature space. At the same time, due to its nature, the algorithm is easily parallelizable, suiting large real-life applications. Not only can it be employed in a wide range of prediction problems but it also has only a few parameters to tune [44]. Random Forest classification has also been engaged in connection with protein language models with successful results [40]. For these reasons, this algorithm seemed a good fit for the phage-host protein interaction prediction assignment.

More precisely, the model implementation used is the one that can be found in the scikit-learn Python library. The parameters tuned in this research and the tested values are the following:

- **n_estimators** - The number of trees in the forest - [50, 100, 150, 200]
- **criterion** - The impurity metric - [Gini]
- **min_samples_split** - The minimum number of samples in a node for it to be split - [2, 3, 4]

- **min_samples_leaf** - A split that would leave less than this amount of samples in any leaf will not be executed - [1, 2, 3, 4]
- **max_features** - The number of splitting features - [log2, sqrt] Note that “None” in the function actually checks all the features

2.6 Nested cross-validation and grid-search

Scoring a model on a simple train-test split basis leads to biased estimations. Hence, cross-validation (CV) is a popular method to evaluate a model’s performance. The dataset is split into K partitions. Then, the model is trained and evaluated K times. Indeed, every partition is used once as a test set, after training the model on the remaining samples. The final model score is the average of the results obtained in each fold [45].

On top of that, tuning of the algorithms’ hyperparameters is often required. In this case, the sklearn’s GridSearchCV method for model selection comes in handy. It takes a set of possible parameters as input and trains the model with each possible combination.

In the project, the grid search was combined with a nested cross-validation. A nested cross-validation adds a layer to a classic CV. In other words, on each partition - or outer fold - is performed a further cross-validation, producing inner folds on which training the model - see Figure X. When evaluating the protein-interaction classifiers, a grid-search was performed for each outer fold, and for each combination of parameters was performed an inner cross-validation. The result is an estimation of each combination’s performance with a reduced bias. The best-performing combination of hyperparameters is then used for the fit on the outer training set.

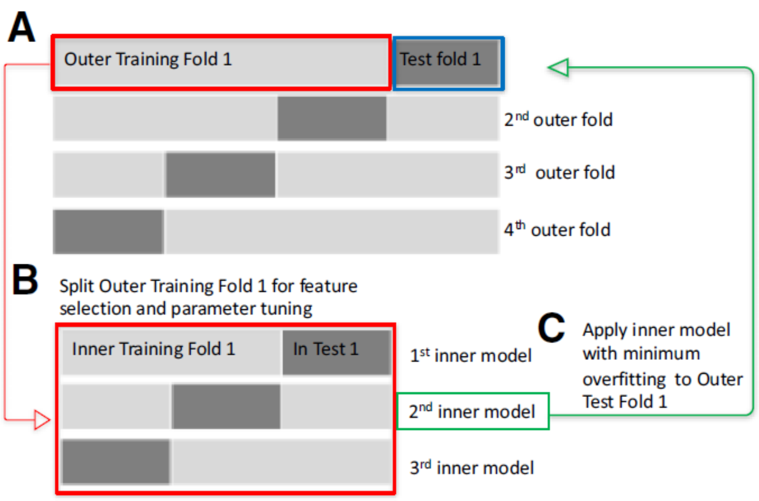


Figure 2.3: Depiction of a nested cross-validation, from [46]. With grid-search, inner cross-validation is performed for each parameter combination to estimate its performance with reduced bias. The best-performing combination is then fit on the outer training set.

Sklearn’s StratifiedKFold cross-validation technique was used. It keeps the labeling distribution in the splits unchanged. The methods here described were used to test every combination of the parameters listed in the classifiers paragraphs above.

2.7 Procedure

Raw datasets had to be elaborated to obtain some data that could be the input for the models performing protein-protein interaction prediction. In a binary classification like this one, the entries labeled as interacting are named positive samples, and the others negative. The goal is to obtain a dataset that associates a phage protein with a bacterial protein. Each protein pair must be labelled as interacting or not More specifically, the focus is on the identification of the interaction between viral receptor-binding proteins (RBPs) and host receptor proteins (RPs).

First of all, in the BASEL receptors set, the receptors were listed in the “primary_receptor” and “terminal_receptor” columns. In each of these lists were present both the lipopolysaccharide (LPS) and proteic receptors. The LPS cannot be used for the following analysis. Thus, a new column was added, listing the bacterial receptor proteins that appeared in either “terminal_receptor” or “primary_receptor”, disregarding the LPS. Some phage entries did not carry information about a protein receptor. On top of that, the samples for which the receptor didn’t correspond to any gene in K12’s proteome were discarded, reducing the number of viruses from 78 to 52. Each of these entries was concatenated with the aminoacidic sequence of the respective bacterial gene, linking the bas identifier to the host protein. The last step to obtain the protein pairs consists in connecting each bas identifier also to the viral sequences in the BASEL proteome. In this dataset, many different proteins belong to each virus, and in principle, they might all be paired with the host sequence. On the other hand, only a limited selection of these possible combinations would interact. The labeling of the protein- pairs is not trivial, since a dataset of experimentally validated interactions between RBPs and bacterial receptors is not available.

The labeling took place in the project and relied on the hypothesis that if a phage infects the bacteria, it must be due to a conjunction between the receptor and a protein located in the distal tail fiber. In other words, the labeling task was reduced to the identification of the correct tail fiber proteins. Firstly, among the set of sequences comprised in the BASEL proteome, a selection was made based on the name reported in the “protein” column. This approach made use of a regex to select the RBPs [40]:

```
tail?(.?|\s*)(?:spike?|fib(?:er|re))|recept(?:o|e)r(.?|\s*)(?:bind|
    recogn).*(?:protein)?|(?<!\w)RBP(?!a)
```

Listing 2.2: Regex for the selection of receptor-binding proteins. It filters the BASEL proteome entries based on the annotation in the ”protein” column

On top of that, a protein would not be included if the name stated one of the following: ’putative’, ’chaperone’, ’assembly’, ’RNA’, ’connector’, or ’baseplate’, no matter the case. The set of positively labeled protein pairs was obtained with an inner join between the filtered BASEL proteome and the updated BASEL receptors, based on the “bas” identifier. A total of 107 positive samples was achieved.

The input dataset for the language and classifier models requires also negative protein-pairs. Three kinds of these could be obtained:

- Non-tail fibers (excluded from the filtering) coupled with any other bacterial protein in the K12 proteome
- Tail fibers with any receptor that is not their partner - for as it is identified in BASEL receptors
- Tail fibers with any K12 protein that is not their partner

A huge number of combinations could be attained this way. However, it would make little to no sense to include them in the dataset, considering the few positive samples available. The final set comprises 1112 points, 107 of which are positive pairs, and the rest are equally distributed among the three negative kinds.

For the modeling part, the data is loaded in a pipeline which is built out of three consecutive components: the language model to produce the embeddings, the oversampling module, and the final classifier. The script can be run specifying the desired component to use in each segment of the pipeline from the command line of the terminal. The embeddings of the aminoacidic sequences are produced sequentially - first, the phage sequences, and then the bacterial - due to huge memory requirements for their parallelization.

The random forest can easily handle protein-level representations. The phage and bacterial embeddings - each of size 1024 - are concatenated side-by-side. The random forest is thus fed vectors of size 1 x 2048. The embeddings are produced once for all the data at the beginning of the pipeline, and they can be saved and loaded the next time the script runs. On the other hand, the resampling and the classification must be performed inside the nested cross-validation. Generating the synthetic samples before arranging the data into folds would result in some identical observations between train and test sets, leading to optimistic performance estimations.

3 Results and Discussion

3.1 Dataset composition

The whole project was developed with one idea in mind: the possibility of using the model as an RBP-RP interaction predictor, given two generic protein sequences as input. This hypothesis contributes to defining the characteristics of the dataset, and the dataset influences the model's performance. With a similar scenario, many more possible combinations would fall in the basket of not-interaction pairs than the other way around. While the positive samples definition criteria are rather straightforward, the protein pairs contributing to the negative set can be open to interpretation. As stated in paragraph 2.7, the wrongly coupled proteins adopted to train the model are of various kinds. The ratio of negative to positive samples is 10:1. However, the ratio is arbitrary, and it must be considered that a more balanced set would likely lead to better testing scores of the classifier, but less capability of generalization in real-world usage. The data imbalance has to be taken into account in the analysis of the results.

3.2 Pipelines compared

The classifier in use here is a Random Forest, and its base component is the Classification Tree - see paragraph 2.5.1. The nature of the aminoacidic data is complex, and a non-linear model like the tree classifier should be a good fit for the interpretation of such input. The combination of two language models - ProtT5 and ProtXLNet - with three resampling techniques - SMOTE, ADASYN, and no resampling - results in the usage of the Random Forest algorithm in six different pipelines. In the following sections, their performances will be confronted. While each version of the embedded sequences was obtained only once, the oversampling technique as well as the Random Forest entered the nested cross-validation loops. It is important to notice that the resampling couldn't happen on the main dataset since it would produce invalid test and validation sets. The reason is double. Firstly, they would not mirror the actual distribution of the initial data. Secondly, due to the nature of the resampling techniques, the synthetic samples would be too similar to the ones seen during the training. It follows that such an application of the oversampling would lead to optimistic performance estimations.

Classifications can produce four different types of predictions:

- **True Positive (TP)** - Positive sample labeled as positive
- **True Negative (TN)** - Negative sample labeled as negative
- **False Positive (FP)** - Negative sample mislabeled as positive
- **False Negative (FP)** - Positive sample mislabeled as negative

Many commonly used metrics for binary classification make use of these values. Some of these are accuracy, precision, recall, and F1 score. Accuracy is the number of correctly classified elements on the total [47]:

$$\text{Accuracy} = \frac{TP + TN}{TP + TN + FP + FN}$$

Precision states the fraction of positive predictions that are true positive samples:

$$\text{Precision} = \frac{TP}{TP + FP}$$

Recall indicates the proportion of positive samples actually labeled as positive:

$$\text{Recall} = \frac{TP}{TP + FN}$$

Finally, F1 is defined as the harmonic mean of precision and recall. However, this formulation is undefined when the number of true positives is 0. For this reason, a more convenient expression of F1 is [48]:

$$F1 = \frac{2TP}{2TP + FP + FN}$$

Since the F1 score collects both precision and accuracy in a single metric, it was used as the target value to maximize during training. A first approach to assess the pipelines' performance is thus the comparison of their F1 scores. In fig. 3.1 the mean scores and their standard deviations across the outer folds of the cross-validation are plotted. On each fold, a distinct model is trained and tested using the best-performing set of parameters selected in the inner folds of the cross-validation. On top of that, the score of the model that best performed in its fold is indicated. The combination of parameters of the best performers in the outer fold of each pipeline is reported in table 3.1.

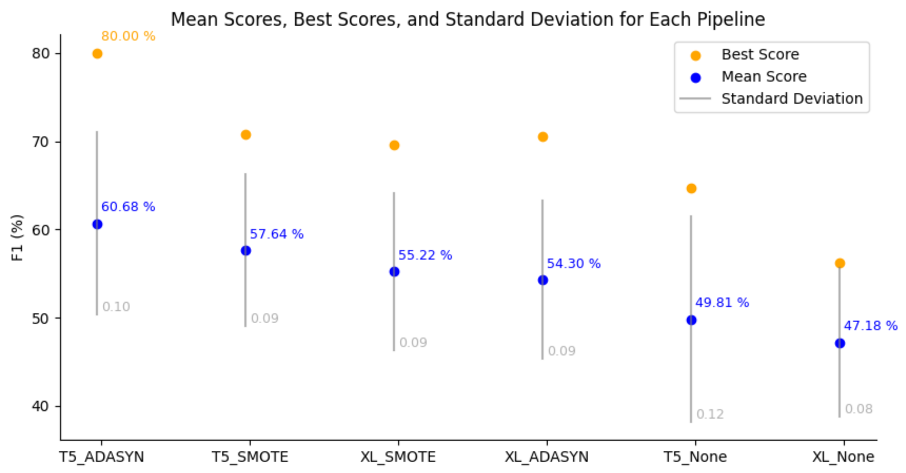


Figure 3.1: F1 scoring of the Random Forest pipelines. The predictions on which the scoring is based are obtained by classifying a sample as positive when the majority of the trees in the Random Forest do so. The blue dots are the mean scores across the outer folds of the cross-validation, while the grey bars are visualizations of their standard deviation. On each fold, a distinct model is trained and tested using the best-performing set of parameters selected in the inner folds of the cross-validation. The yellow dots symbolize the score of the model that performed best in its outer fold. The percentages indicate the mean F1 score.

Among the tested settings, the best performer was the classifier which relied on ProtT5 embeddings and ADASYN resampling. It performed best not only on average across the outer folds, with a mean F1 score of 60.68 %. Not by chance the best combination of parameters in this circumstance achieved an F1 of 80 %. In previous works, ProtT5 had been reported already as able to encode protein features starting from the aminoacidic sequence [40]. The present results seem to confirm it as the most valuable model. As a matter of fact, among the pipelines including resampling, ProtT5 achieved higher average

Classifier	max features	min samples leaf	min samples split	n estimators
T5_ADASYN	log2	3	2	100
T5_SMOTE	sqrt	4	2	200
XL_SMOTE	log2	4	2	150
XL_ADASYN	sqrt	2	4	200
T5_None	sqrt	2	2	100
XL_None	sqrt	3	3	200

Table 3.1: Parameters of the best-performing models in the outer fold of the cross-validation, for each pipeline. The score achieved in their respective fold is plotted in fig. 3.1 as a yellow dot. In every case, the ”criterion” parameter is ”Gini”. Not reported for table’s clarity. The pipelines are sorted in order of F1 score.

scores than ProtXLNet. Conversely, when no oversampling is actuated, the model suffers a lack of examples of truly paired proteins. The models trained on the various outer folds have similar oscillations in their results. Indeed, the standard deviation always settles around 0.10. The best overall parameters - i.e. the best ones of T5_ADASYN, the first row of table 3.1 - were used to train a new model across all the outer folds. This way, the model could be trained on the whole dataset, instead of having to split it with other versions of the same pipeline. Anyway, the average result wouldn’t change much, confirming a similar score of 60.37 %, and a best occurrence at 72.34 %, and a standard deviation of 0.07. Different results might arise from two main actions: the exploration of a broader space of hyperparameters, or a dissimilar engineering of the training dataset.

One step further in understanding the pipelines’ diverse behaviors is the visualization of precision, recall, and accuracy as separate entities - fig. 3.2. Here is evident how the deeply different datasets among resampled or not influenced the learning strategy of the models. The T5_None and XL_None systems expect to see only a few positive samples and conduct a stricter judgment. They have thus a rather high precision - 67 % and 72 % for XL_None and T5_None, respectively. Anyway, reducing the number of positive assignments comes at the cost of an eroded recall of the valid interactions, that cannot overcome 40 %. In contrast, the models that embraced the knowledge of a dataset with a more balanced distribution are looser when it comes to testing real data, with a precision of 56-58 %. The bright side of reduced precision is the higher recall, which peaks at 70 % with T5_ADASYN, the best classifier. The accuracy is quite constant and elevated. However, it should not be trusted, as it is influenced by the composition of the dataset.

3.3 About thresholds

The majority vote among the trees of the Random Forest determines the predicted class among the observations. To put it another way, the highest fraction - relative to the total number of outcomes of the algorithm - specifies the output. To clarify once again, a protein pair is deemed interacting if according to the Random Forest, its probability to be positive is greater than or equal to 50%. Nevertheless, it is not the only way to split observations. Moving the threshold above or below the middle line changes the amount of positive and negative assignments. It follows that the true positives, true negatives, false positives, and false negatives will modify, altering precision, recall, and F1 accordingly. A model must assign high probabilities of positive classification to actually positive samples, and as low as possible to the negatives. Once this is done, a threshold could split the two categories into clean sets. A direct visualization of this is a distribution plot, showing the

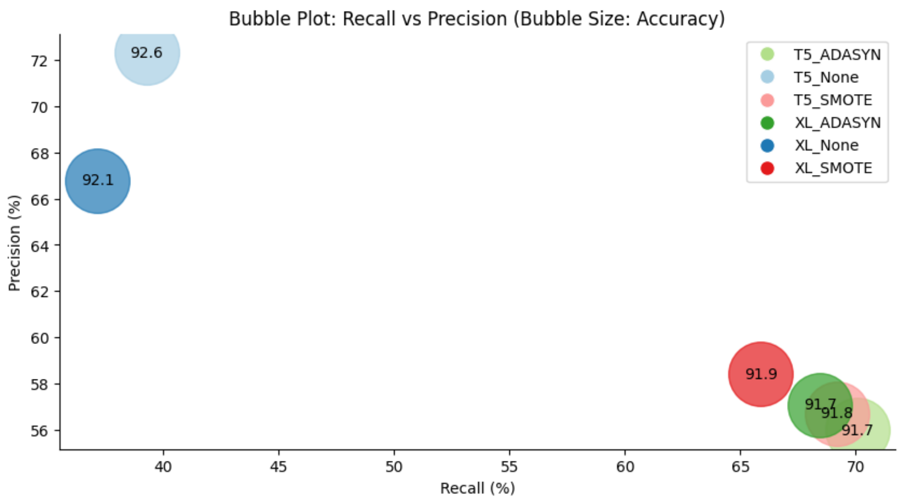
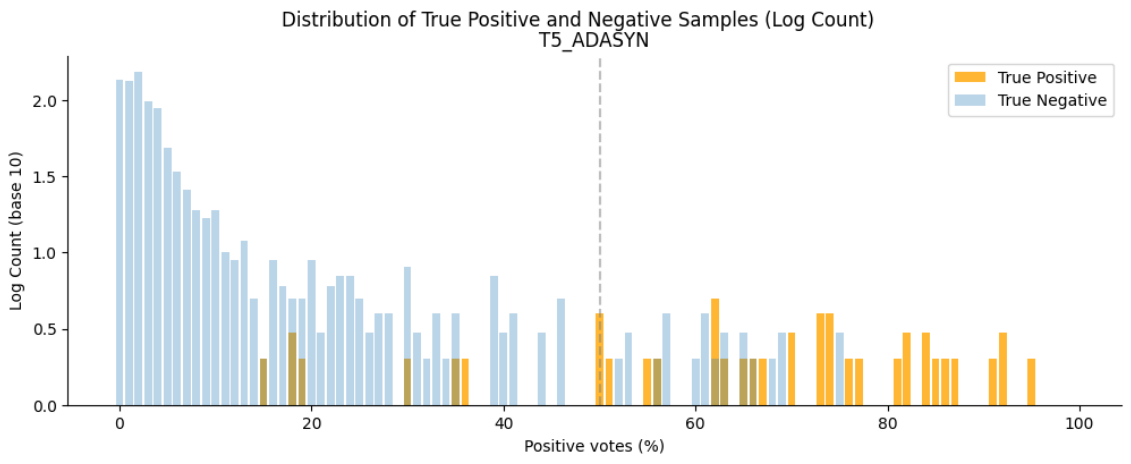
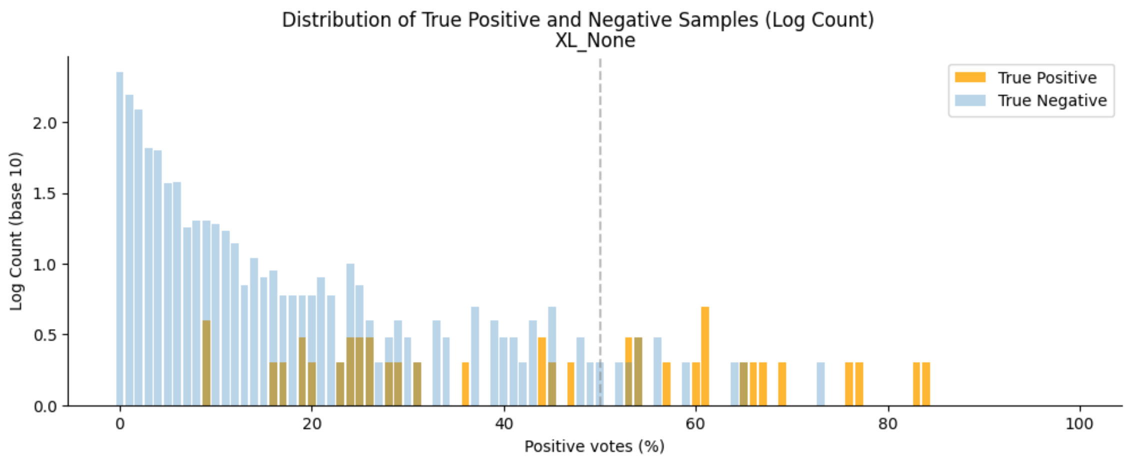


Figure 3.2: Bubble plot representing the mean precision (Y-axis), accuracy (X-axis), and recall (bubble size) of the pipelines. Green, blue, and red denote respectively ADASYN, None, and SMOTE resamplings. The ProtXLNet language model is in full colors and ProtT5 is in paler tones. From now on, this color code will be kept.

number of both classes of observation for each estimated probability. The distributions of T5_ADASYN and XL_None are shown in fig. 3.3. The negative samples richly outnumber their counterpart, so the counts are on a log_{10} scale to appreciate also the smaller set. The samples originally categorized as negative are depicted in blue, and the others in orange, while a hypothetical 50 % splitting threshold is shown as a dashed line. Here is rather easy to comprehend the role of the aforementioned metrics. The true negatives are all the blue observations that fall to the left of the line and the true positives are the orange ones on the other side. Conversely, the false negatives are the orange sample on the left side of the split, and the false positives are the blues to the right. The precision value denotes the number of true positives among all the instances that lie above the line. Recall, on the other hand, is the proportion of the positive samples on the right side of the boundary. Unless the model was not able to perform any separation, moving the threshold higher - i.e. a stricter classifier - increases the precision. Inevitably, at the same time, more positives will fall on the other side, reducing the recall. Lastly, accuracy represents the ratio of observations on the correct part of the X-axis, over the total. It is clear now how all the models reach top accuracy: the vast mass of negative samples will effortlessly be parted correctly, inflating the metric value. If the threshold was set at 100 %, the models would still show a top-level accuracy. Precision, recall, and F1 are better tools for model evaluation when the underlying distribution of the samples is unequal. The best and the worst models - namely T5_ADASYN and XL_None - are compared in fig. 3.3. It is immediate how the first operates a better separation among the two distributions. In these conditions, it is possible to find a threshold that isolates observations on the appropriate territory. Simply eye-balling the plot, 50 % or a little less seems a good candidate. Examining the other face of the coin, XL_None overlaps the two distributions to a large extent, making the splitting impossible. Furthermore, the overall dataset is shifted to the right, mirroring the easier tendency to deem observations as non-interacting learned from a skewed training set. The setting of the proper threshold depends on the desired application of the predictor. The context defines which one between precision and recall weighs more on the scales.



(a) T5_ADASYN



(b) XL_None

Figure 3.3: Distributions of the truly-labeled samples across positive-labeling probabilities for T5_ADASYN (3.3a) and XL_None (3.3b). The count is on a log_{10} scale. Blue: negative samples. Orange: positive samples. The dashed line embodies the 50 % splitting threshold.

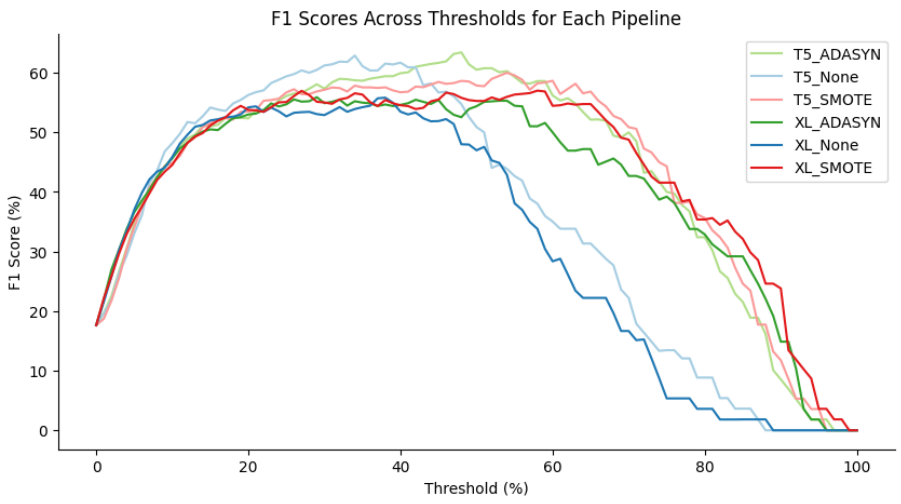


Figure 3.4: F1 score versus probability threshold for each pipeline. The color code is as in the previous pictures. Each point is calculated as if the splitting line in fig. 3.3 was set at different values along the X-axis.

T5_ADASYN's superiority on matters of F1 score has been confirmed already for the standard boundary. Nevertheless, analyzing these results on moving thresholds can give further insights. F1 is not possible to extract from the distributions, so it's better to plot its calculated value as a function of the probability threshold. Such a representation can be seen in fig. 3.4 for each pipeline. Once again, the diverse behavior of the pipelines without oversampling is obvious. In general, all the curves top at 60 % F1 or slightly below, but T5_None and XL_None start dropping after the middle threshold. The explanation lies on the left-shifted distribution of the positive samples they produce. Sure enough, the scarcity of true positives reduces the F1 value. The same observation is true also for the other models but at stricter boundaries. All the curves reach a 0 score between 95 % and 100 %. Said differently, no samples are predicted as positive with a close-to-absolute certainty. Echoing the ponderings made on fig. 3.1, the ProtT5 versions are noticeably more favorable than their companions, keeping consistently enhanced performance over ProtXLNet for every splitting criteria. The ADASYN technique seems to influence the outcome to a greater extent than SMOTE, inducing a wider gap between the T5 and the XLNet arrangement. The anticipated apex of T5_ADASYN at 60 % F1, located on the 50 % threshold circa is the top general score. Noteworthy, T5_None obtained a similar achievement with a threshold below 40 %. This arises quite certainly from the positive samples being included to a higher extent, meaning a raised recall. Specifically, T5_ADASYN has the highest F1 of 63.41 % with a threshold of 48 %, and T5_None the second highest, 62.88 % with a threshold of 34 %.

Classification models are frequently evaluated by adopting the Receiver operating characteristic curve (ROC). It plots the true positive rate (TPR) against the False positive rate (FPR). The TPR is another name for the recall. The false positive rate is:

$$FPR = \frac{FP}{TN + FP}$$

To visualize the concepts, TPR and FPR are respectively the proportions of the positive or negative distribution that lies above the probability boundary. Each curve originates from the combination of TPR and FPR at various probability thresholds. The ROC curves of the pipelines are plotted in fig. 3.5. In the graph, the threshold is set higher close to the origin and lowered toward the edges. Less strict criteria include more samples as positives, raising both TPR and FPR. A random model does not separate the underlying populations at all, resulting in equal TPR and FPR. Such a model is usually set as the benchmark baseline and is represented as a diagonal in the ROC plot. On the other side, a perfect model would operate a perfect separation, meaning that at each split configuration, the TPR would be 100 % and the FPR 0 %. Such a scenario manifests as a curve that adheres to the top left corner. The closer a ROC curve gets to this shape, the better the model. Comparing the curves requires a punctual metric. As such, the area under the curve (AUC) is calculated. The hereby-trained models have optimal shapes and AUC values, consistently around 93 %. Sadly, all that glitters is not gold. Once again, the imbalanced dataset invalidates this classical validation. The high number of non-interacting protein pairs shrinks the false positive rate, resulting in a left-shifted curve.

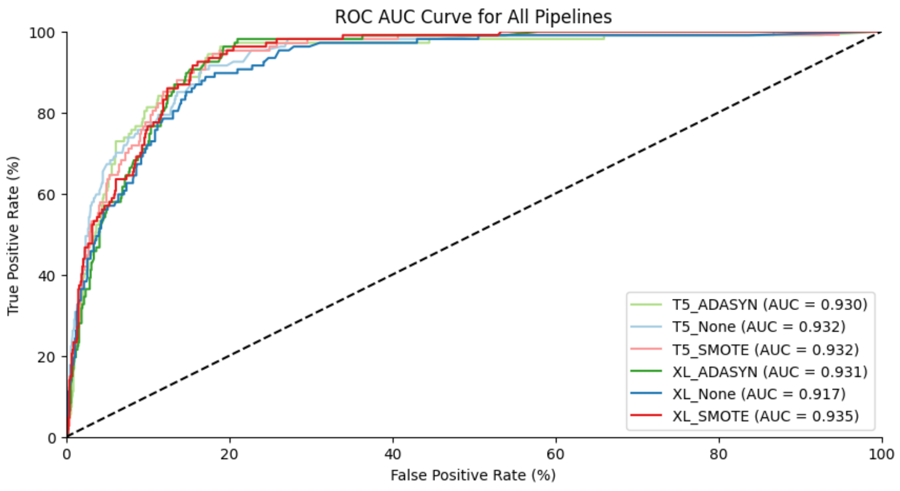
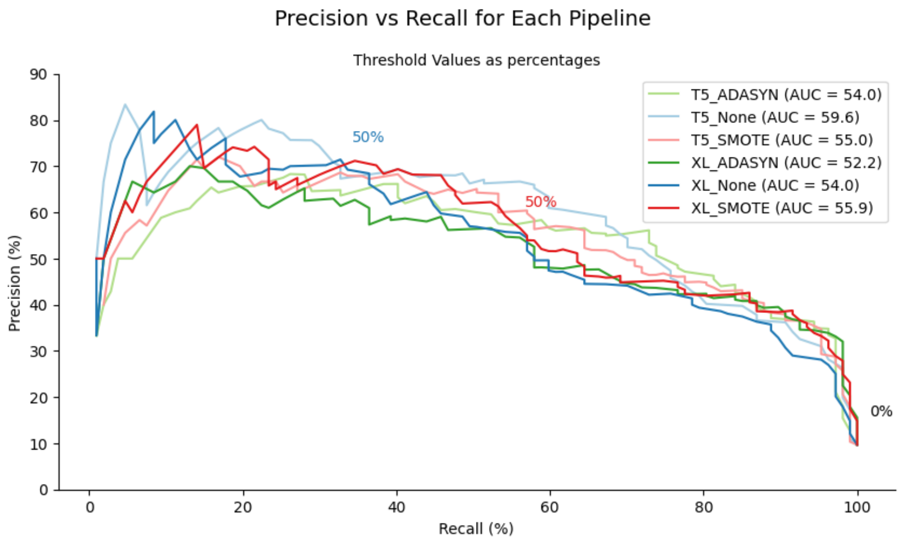
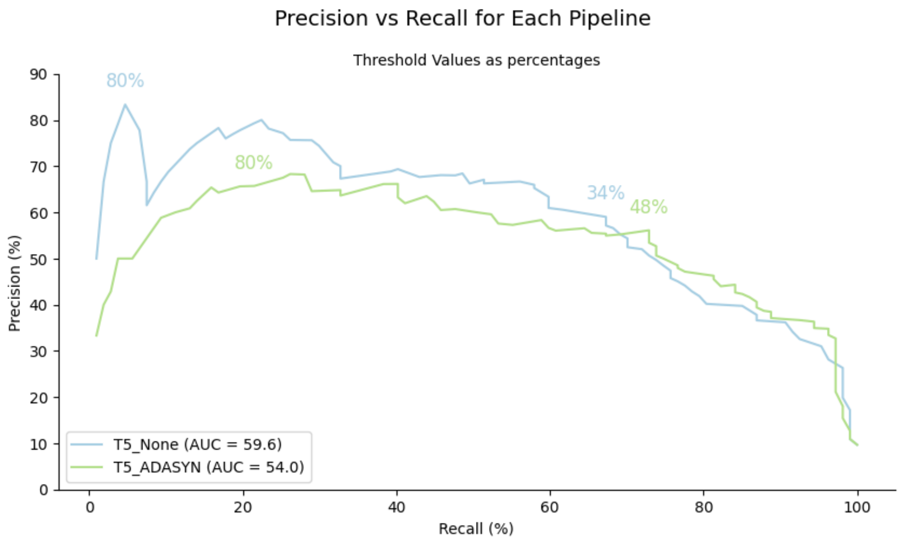


Figure 3.5: ROC AUC curve of all the pipelines. The diagonal dashed line represents the performance of a random classifier. The area under the curve is stated in the legend.

An alternative way of performing this analysis is the precision-recall curve. The absence of spoiled false positive rates makes it more suitable for the situation. The plot in question is portrayed in fig. 3.6. Here the results are drastically more realistic. It must be noted that the points of different pipelines laying on the same X coordinate don't correspond necessarily to the same threshold. Indeed, in fig. 3.6a, the middle split value is reported for XL_SMOTE and XL_None as representative of the two main groups of models: oversampling or not. Again, it can be seen as at the same threshold, the latter have a lower recall than the former, confirming what is seen in fig. 3.2. The two pipelines that achieved the top F1 scores - T5_None and T5_ADASYN - are compared in fig. 3.6b to get some more insights on the nature of the score. The threshold corresponding to the summarizing value - 34 % and 48 % stay quite close to each other on the graph, denoting a similar split performed in the two cases. More precisely, the oversampled model has a precision of 56.12 % and a recall of 72.90 %. The other has slightly higher precision and lower recall, respectively 59.02 % and 67.29 %. The AUC values of T5_None are more than 5 points higher than T5_ADASYN's, and its curve is mostly higher. However, these indications can be tricky. T5_None's thresholds are always more left, indicating a decided tendency to reduce recall very quickly. Selecting the correct threshold might lead to appreciable results in both cases and maybe even specific applications concerning precision and accuracy preferences. On the other hand, resampled models' performances seem more stable and trustworthy. If the dataset was expanded, the behavior of newly-trained ProtT5 and ProtXLNet forests without oversampling might lead to less predictable outcomes.



(a) All pipelines



(b) T5_None and T5_ADASYN

Figure 3.6: fig. 3.6a: Precision versus recall for each pipeline across the thresholds. As a tendency, the threshold starts high close to the origin and lowers toward the edge. Anyway, not all the curves have the same threshold on the same ordinate or abscissa. The 50 % threshold for XL_SMOTE and XL_None are reported as a reference for oversampled and not pipelines. fig. 3.6b: Only T5_None and T5_ADASYN are shown.

3.4 Recall the hand-crafted features

The large language models are a recent technology, and uncountable applications are waiting to be explored. In this project, this deep-learning technology is used in combination with more classical machine-learning methods to predict if two proteins can interact, with knowledge restricted to their protein sequences. Other attempts on similar tasks have recently been made. For example, in 2021 Boeckaerts et al. [49] managed to make predictions about the bacterial host of a phage, on the species level. Various algorithms were employed to enact the classifications, but not the language models. Various classifiers were tested, with the Random Forest selected as the best performer. Instead of the sequences, the input was a set of key features individually calculated from DNA and protein sequences. Summing up, the input is more easily interpretable than the whole aminoacidic series but requires previous work, and the classification is not on the protein-protein level, but on the more generic protein-host. One more difference lies in the classification: it attempts a multiple-class categorization, in contrast to a binary split. The next ring in the chain is the production of Gonzales et al. [40]. The large language models were introduced in the host prediction task, switching to aminoacidic sequences as input. The classification was brought one step backward on the level of detail. The forecasted phylogenetic element is the genus, which encompasses the species. Several language models were employed, with a Random Forest as the final classification algorithm. The project takes up from here, in the attempt to continue on the path of the sequential inputs, but increase detail to the protein-protein level. Further comparisons between the results and approaches of each of these projects and the current work are found in the following paragraphs. The first comparison is with Boeackaert et al. [49]. The dataset was constructed by initially collecting 1170 RBP sequences from various sources. Keywords like ”Tail fiber” and ”Tail spike” were used to identify the molecules. The collected proteins are related to bacterial hosts. Some species are more represented than others. For example, *E. coli* appears in one-third of the set. After the removal of low-numbered species, the sample size was reduced to 887. Seven different categories could be predicted. The feature space includes attributes like the Guanine and Cytosine content of the DNA string, the frequency of each amino acid, the molecular weight of the protein, and many other chemical-physical qualities for a total of 218 attributes. Four different models were trained, adopting a nested cross-validation to fine-tune their parameters. Furthermore, the proteins were grouped according to the similarity of their sequences, aiming to reduce the redundancy across the folds. The Random Forest was selected as the best classifier. Its precision-recall plot is reported in fig. 3.7. In the plot, there is one curve for each grouping criterion. If compared with fig. 3.6, the statistics are noticeably better, with more extended areas under the curve. The formulation of the prediction problem is different though, and a confrontation would not be correct. However, some considerations can be made. First of all, the lower the grouping threshold, the lower the scores. If in the 100 % similarity group the bacteria worst recognized by the classifier is associated with a 76.6 % accuracy, the 50 % similarity comes with a more drastic 30.4 %. In this project, no kind of grouping was done. It is possible that a clustering aimed at reducing the redundancy of the sequences could lead to improved results.

Among the biological features employed in the research, just a few stand out as relevant for the prediction. First, the Z-scale descriptor, a measure of amino acids lipophilicity. A lipophilic molecule dissolves better in lipids than in water. Given the lipidic nature of the bacterial membrane, we can speculate that different lipidic affinities might be required to ease access to heterogeneous hosts. Second, the frequency of the positively charged amino acids arginine and lysine. Lastly, the guanine-cytosine content. This measure cannot be obtained from the protein sequence. Guanine and cytosine are two of the four nitrogenous bases making the DNA. Each amino acid can be encoded by multiple combinations of three bases, making it impossible to obtain this information directly from the protein string. Such value is usually used to gain insights into the evolutionary distance between two organisms. Nevertheless, the more a species evolves, the more mutations occur, possibly modifying the GC content. This fact might need consideration. To construct the dataset for this thesis, a necessary implication was necessary: a protein pair

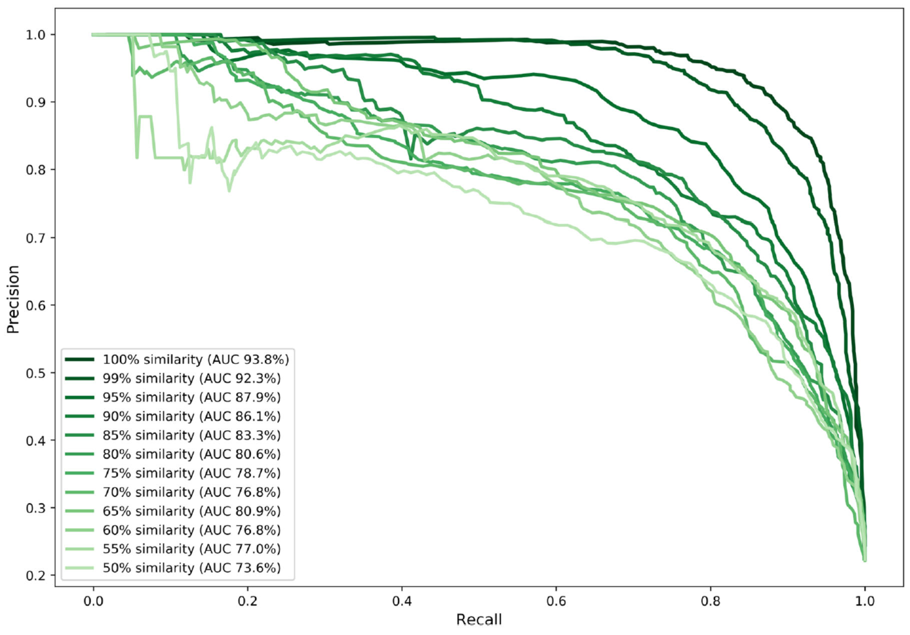


Figure 3.7: Precision-recall curve from Boeckaerts et al. [49]. Each curve originates from a different grouping of the sequences. The sequence similarity reported in the legend refers to the criteria for grouping the proteins. The area under the curve is also noted in the legend.

is deemed as interacting if and only if there is infection. However, this is not necessarily the full story. Two proteins might bind, but the following steps of the infection could not succeed for various reasons. One of these reasons is the genomic dissimilarity between the virus and the host. If the infection is not observed the protein pair is labeled as not interacting, introducing some noise in the model's dataset. A possible approach to exploit this feature is the embedding of the DNA coding sequences, instead of their translated version.

3.5 It takes confidence

The comparison with the job of Gonzales et al. [40] is more direct. Their pipelines involve seven different language models in combination with a Random Forest, in a setting parallel to the one developed here. The focus is to understand if the host prediction is possible, and which embeddings are the most suitable. Additionally, the amino acids embeddings were averaged to the protein level - see section 2.4.1. The main differences stay in the usage of one-only sequence as input in contrast with two, and the multiple nature of the classification instead of binary. The dataset is composed of RBP sequences, obtained via a regex filtering of the proteomes of 15,823 phages with host annotations. The final formulation sees 24,752 RBPs across 9,583 phages and 232 hosts. The 96 % of the entries are associated with only 25 % of the hosts, resulting in an imbalanced set. In a first attempt at balancing the data, the remaining 75 % of bacterial genera was removed from the main dataset and was reincluded only in the test to mirror a realistic scenario. Here, they are clustered together under the label ”other”. Resampling techniques like SMOTE and ADASYN were not adopted. However, in a second pursuit of an even data distribution, higher misclassification penalties were applied to minority classes. The training took place under a stratified nested cross-validation for parameters' tuning, with the F1 as the target metric. To make this thesis more comparable, the parameter space mirrors the one of

Gonzales et al. - see section 2.5.1. The optimal set resulted to be:

- **max_features**: sqrt
- **min_samples_leaf** : 1
- **min_samples_split** : 2
- **n_estimators** : 150

The previous section of this report makes extensive use of the positive probability threshold. In a multiple-class configuration, the confidence of the estimations must be expressed in another way. Gonzales et al. define it as the difference between the labeling probabilities of the two most expected genera. To put the analysis on the same plane, a correspondent measure is hereon adopted. In the binary classification, this confidence is:

$$k = p - (1 - p) = 2p - 1$$

Where p is the positive probability, and $(1 - p)$ is the negative classification probability. Both in the binary and in the multiple classification, if the outcome confidence is lower than a set threshold, the sample is put in the ”others” group. This thesis project does not include any ”others” group. It follows that any prediction below the threshold is simply mislabeled. The F1 scores across these thresholds are confronted in fig. 3.8. In the plot are depicted the results relative to the ProtT5 and ProtXLNet. One additional set of results appears. Gonzales et al. used the model of Boeckaerts et al. [49] - trained on their dataset - as a benchmark. It is recorded also in the graph. The corresponding lines are short due to the limited range of confidence threshold stored in the reference. The first thing that catches the eye is the rapid drop of the non-balanced classifiers' scores. This new point of view exposes the inconsistency of their predictions. But not only. Also, the other models employed in this thesis, which seemed to have acceptable performances so far, can't hold a candle to the estimators in the references. In the Gonzales et al. context, ProtT5 is consistently better than ProtXLNet, and they both surpass the benchmark. They all obtain better results than T5_ADASYN, T5_SMOTE, XL_ADASYN, and XL_SMOTE, which in turn leave T5_None and XL_None as the less reliable predictors.

3.6 Biological matters

Large language models combined with a Random Forest classifier can give reliable estimations of a host genus based on the sole sequence of a phage protein. The amino acids seem to carry a good amount of information to understand what is the possible target of the virus. On the opposite corner, the same experiment settings are not satisfactory in probing the communication between two macromolecules. The information encoded by each protein with these methods could not be the correct kind to acknowledge the interaction. The sequence determines the three-dimensional structure, and the final conformation - as well as the aminoacids properties - determines the relation between receptor and ligand. More simply, the string has physical-chemical characteristics. The embeddings might carry the latter information, without thorough details on the tertiary structure. Knowing which areas of the proteins are exposed to the surface, and what polarity, electric charge, or size they have, could be enough to define the target genus. Bacteria belonging to the same genus have a similar structure, if compared with the other phylogenetic branches. Such structures must have similar weaknesses to viral proteins. The variety of lipids that constitute the membrane, as well as the nature of the lipopolysaccharide, would enable

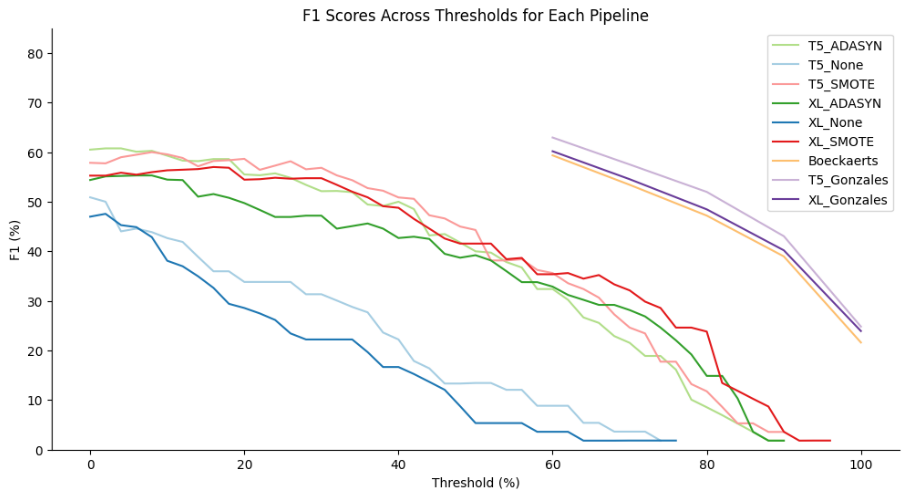


Figure 3.8: F1 scores across multiclass confidence thresholds. Light purple: the Random Forest of Gonzales et al. [40] with ProtT5 embeddings. Deep purple: the Random Forest of Gonzales et al. with ProtXLNet embeddings. Orange: Gonzales et al. adaptation of Boeckaerts et al. [49] model. The other colors are consistent with the previous color code. The range of the curves is limited to the data reported in the reference.

the access of external molecules with specific features. Conversely, the specific interaction between two proteins has a really precise nature. Not only the polarity and charge should be correct, but also the size and shapes complementary. It requires a detailed knowledge of molecule conformations to make such a forecast.

The story might be even more complex. Here the focus is on proteins and their aminoacidic blueprint. However, genomes carry important information. In the previous section, the GC content was considered, and how the DNA similarity between phage and bacteria could determine the successful infection, introducing some bias in the labeling of protein-protein interactions. On top of that, the nucleic acids - inside the protein gene or elsewhere in the chromosome - determine post-translational modifications. These reactions introduce variety to the protein in its maturation process, changing its properties. The amino acid knowledge does not necessarily carry information about them. Lastly, the molecules don't exist in an empty space, they are immersed in some matrices or solutions. The environment influences the conformation and behavior of the proteins. For example, the pH and temperatures must usually be in a narrow range to allow proper functioning. These factors introduce ulterior variation in the protein's final structure and attitude to interact.

3.7 Space for improvement

The accepted protein input, the dataset creation and labeling criteria, the sequence encoding method, and finally the classification algorithm all contribute to the quest's success. The exact combinations attempted here brought partial results, and the Random Forests could not separate the distributions efficiently. Many more attempts wait to be made, starting with the initial question. Instead of "Does this pair of phage and bacterial protein interact?" it could be "Does this receptor-binding protein interact with a possible hosts protein?", or "RBP and RP", or "Do these DNA strings originate interacting proteins?" and on, and on. Every different question sets a different starting point and selects sample populations from the real world. Also, the only considered bacterium here is *E. coli K-12 MG1655 ΔRM*. Exploring a broader range of species and strains might give access to improved learning. The next step is the labeling of interactions. So far, the most frequent approach seems to rely on filtering the annotations in protein databases. However, more accurate techniques might exist for the determination of a protein type. Additionally, when are two proteins deemed interacting? Here, a viral product categorized as RBP is considered a ligand to a bacterial receptor protein, if that bacterium is a known phage target. However, the acquisition of experimental confirmations on binding molecules would provide a more solid ground for the analysis. Moreover, the dataset imbalance played a major role in the learning process. The number and type of negative samples included would alter the splitting of their distributions. An even amount of positives and negatives in the starting data might help the model's adaptation, but be less stable in a real-world application, and vice-versa.

The next matter concerns the sequence encoding technique. Large language models are the current state-of-the-art to project literal features to a numerical space. Specifically, ProtT5, ProtXLNet, as well as other transformer architectures seem to do a good job of automating the acquisition of features that were previously handcrafted. However, their current versions might have learned to reproduce some qualities that are not the most important for comparing two proteins. Some fine-tuning of these models is quite likely to noticeably improve the final predictions. They have been trained on single molecules, while novel arrangements engineered to receive coupled sequences as input might extract the key properties that define their interaction. On another note, proper information about the 3D structure would for sure allow easier predictions, but it comes at a huge computational cost.

Lastly, the possibilities for machine learning algorithms are vast. Random Forests represent a valid choice, but the input information must be suitable. In the current project, protein-level encodings of the two proteins were fed to the algorithm side-by-side. Each one is a fixed length vector of size 1024. Yet other options are available to produce a more compact sample. For example, the two embeddings might be summed or multiplied element-wise, producing a synthesis. Nevertheless, the protein-level averaging causes some information loss from the starting per-residue encoding. The latter sequence conception is as a bidimensional vector. In this scenario, other algorithms might be more suitable. For example, deep learning approaches like self-attention and convolutional networks are promising.

4 Concluding remarks

Are aminoacidic sequences enough to predict two the connection between a receptor and its ligand? Kind of. Proteins from a selection of bacteriophages were coupled with surface receptor proteins of their host *Escherichia coli K12*. The strings were encoded using ProtT5 and ProtXLNet transformers, and classified with a Random Forest algorithm, whose parameters were tuned with a nested cross-validation grid-search. Even though the employment of resampling strategies noticeably improved the performance, the results don't compete with the ones relative to host-prediction tasks. The protein-protein complex recognition seems to require a deeper level of information, and maybe more advanced techniques.

The most promising and accessible opportunities to try next regard the encoding techniques and the classification paradigm. The former lies in the fine-tuning of existing large language models as a source of improvement, the latter in the adoption of deep learning techniques that make use of residue-level information, like convolutional neural networks and self-attention.

Bibliography

[1] Paolo Federico. *GitHub thesis repository*. Accessed on January 27, 2024. 2023. URL: https://github.com/Drivahah/Thesis.

[2] Lenos Archer-Diaby. “Lessons learned from a global history of pandemics”. In: *EMJ Microbiology & Infectious Diseases* 1.1 (2020), pp. 38–41.

[3] William L Langer. “The black death”. In: *Scientific American* 210.2 (1964), pp. 114–121.

[4] Howard Gest. “The discovery of microorganisms by Robert Hooke and Antoni Van Leeuwenhoek, fellows of the Royal Society”. In: *Notes and records of the Royal Society of London* 58.2 (2004), pp. 187–201.

[5] The Museum Of Jurassic Technology. *The life and work of Athanaseus Kircher*. Accessed on January 26, 2024. 2001. URL: https://www.mjt.org/exhibits/kircher.html.

[6] Wikipedia contributors. *Louis Pasteur — Wikipedia, The Free Encyclopedia*. https://en.wikipedia.org/w/index.php?title=Louis_Pasteur&oldid=1197838491. [Online; accessed 26-January-2024]. 2024.

[7] Wikipedia contributors. *Robert Koch — Wikipedia, The Free Encyclopedia*. https://en.wikipedia.org/w/index.php?title=Robert_Koch&oldid=1198213271. [Online; accessed 26-January-2024]. 2024.

[8] Niels Høiby. “Pandemics: past, present, future: that is like choosing between cholera and plague”. In: *Apmis* 129.7 (2021), pp. 352–371.

[9] Kathrin I Mohr. “History of antibiotics research”. In: *How to Overcome the Antibiotic Crisis: Facts, Challenges, Technologies and Future Perspectives* (2016), pp. 237–272.

[10] Graham F Hatfull, Rebekah M Dedrick, and Robert T Schooley. “Phage therapy for antibiotic-resistant bacterial infections”. In: *Annual Review of Medicine* 73 (2022), pp. 197–211.

[11] Marianne Frieri, Krishan Kumar, and Anthony Boutin. “Antibiotic resistance”. In: *Journal of infection and public health* 10.4 (2017), pp. 369–378.

[12] DG Larsson and Carl-Fredrik Flach. “Antibiotic resistance in the environment”. In: *Nature Reviews Microbiology* 20.5 (2022), pp. 257–269.

[13] Sarah F McGough et al. “Rates of increase of antibiotic resistance and ambient temperature in Europe: a cross-national analysis of 28 countries between 2000 and 2016”. In: *Eurosurveillance* 25.45 (2020), p. 1900414.

[14] Benjamin Russell Lewis et al. “Conformational restriction shapes the inhibition of a multidrug efflux adaptor protein”. In: *Nature Communications* 14.1 (2023), p. 3900.

[15] Karanbir S Pahil et al. “A new antibiotic traps lipopolysaccharide in its intermembrane transporter”. In: *Nature* (2024), pp. 1–6.

[16] Laura Michelle Streicher. “Exploring the future of infectious disease treatment in a post-antibiotic era: A comparative review of alternative therapeutics”. In: *Journal of Global Antimicrobial Resistance* 24 (2021), pp. 285–295.

[17] Juliano Bertozzi Silva, Zachary Storms, and Dominic Sauvageau. “Host receptors for bacteriophage adsorption”. In: *FEMS microbiology letters* 363.4 (2016), fnw002.

[18] Emma L Farquharson et al. “Evaluating Phage Tail Fiber Receptor-Binding Proteins Using a Luminescent Flow-Through 96-Well Plate Assay”. In: *Frontiers in microbiology* 12 (2021), p. 741304.

[19] Chandrabose Selvaraj and Sanjeev Kumar Singh. “Phage protein interactions in the inhibition mechanism of bacterial cell”. In: *Biocommunication of phages* (2020), pp. 121–142.

[20] G Schreiber. “Protein–protein interaction interfaces and their functional implications”. In: *Protein–protein interaction regulators* (2020), pp. 1–24.

[21] Mohammed Harris. “Machine Learning and Artificial Intelligence for Pathogen Identification and Antibiotic Resistance Detection: Advancing Diagnostics for Urinary Tract Infections”. In: *BioMed* 3.2 (2023), pp. 246–255.

[22] Abhishek Padhi et al. “Transforming clinical virology with AI, machine learning and deep learning: a comprehensive review and outlook”. In: *VirusDisease* 34.3 (2023), pp. 345–355.

[23] Charles R Harris et al. “Array programming with NumPy”. In: *Nature* 585.7825 (2020), pp. 357–362.

[24] Wes McKinney et al. “pandas: a foundational Python library for data analysis and statistics”. In: *Python for high performance and scientific computing* 14.9 (2011), pp. 1–9.

[25] Fabian Pedregosa et al. “Scikit-learn: Machine learning in Python”. In: *the Journal of machine Learning research* 12 (2011), pp. 2825–2830.

[26] Guillaume Lemaître, Fernando Nogueira, and Christos K Aridas. “Imbalanced-learn: A python toolbox to tackle the curse of imbalanced datasets in machine learning”. In: *Journal of machine learning research* 18.17 (2017), pp. 1–5.

[27] Fernando Pérez and Brian E Granger. “IPython: a system for interactive scientific computing”. In: *Computing in science & engineering* 9.3 (2007), pp. 21–29.

[28] DTUCompute. *Batch Jobs under LSF 10*. Accessed on January 22, 2024. 2023. URL: https://www.hpc.dtu.dk/?page_id=1416.

[29] DTUCompute. *GPU nodes*. Accessed on January 22, 2024. 2023. URL: https://www.hpc.dtu.dk/?page_id=2129.

[30] Enea Maffei et al. “Systematic exploration of Escherichia coli phage–host interactions with the BASEL phage collection”. In: *PLoS Biology* 19.11 (2021), e3001424.

[31] UniProt. *UniProt*. Accessed on January 22, 2024. 2024. URL: https://www.uniprot.org/proteomes/UP000000318.

[32] “UniProt: the universal protein knowledgebase in 2023”. In: *Nucleic Acids Research* 51.D1 (2023), pp. D523–D531.

[33] Alberto Fernández et al. “SMOTE for learning from imbalanced data: progress and challenges, marking the 15-year anniversary”. In: *Journal of artificial intelligence research* 61 (2018), pp. 863–905.

[34] Nitesh V Chawla et al. “SMOTE: synthetic minority over-sampling technique”. In: *Journal of artificial intelligence research* 16 (2002), pp. 321–357.

[35] Haibo He et al. “ADASYN: Adaptive synthetic sampling approach for imbalanced learning”. In: *2008 IEEE international joint conference on neural networks (IEEE world congress on computational intelligence)*. Ieee. 2008, pp. 1322–1328.

[36] Thomas Wolf et al. “Huggingface’s transformers: State-of-the-art natural language processing”. In: *arXiv preprint arXiv:1910.03771* (2019).

[37] Colin Raffel et al. “Exploring the limits of transfer learning with a unified text-to-text transformer”. In: *The Journal of Machine Learning Research* 21.1 (2020), pp. 5485–5551.

[38] UniProt. *UniRef*. Accessed on January 22, 2024. 2023. URL: https://www.uniprot.org/help/uniref.

[39] Ahmed Elnaggar et al. “Prottrans: Toward understanding the language of life through self-supervised learning”. In: *IEEE transactions on pattern analysis and machine intelligence* 44.10 (2021), pp. 7112–7127.

[40] Mark Edward M Gonzales, Jennifer C Ureta, and Anish MS Shrestha. “Protein embeddings improve phage-host interaction prediction”. In: *bioRxiv* (2023), pp. 2023–02.

[41] Leo Breiman. “Random forests”. In: *Machine learning* 45 (2001), pp. 5–32.

[42] Ye Yuan, Liji Wu, and Xiangmin Zhang. “Gini-impurity index analysis”. In: *IEEE Transactions on Information Forensics and Security* 16 (2021), pp. 3154–3169.

[43] Steven J Rigatti. “Random forest”. In: *Journal of Insurance Medicine* 47.1 (2017), pp. 31–39.

[44] Gerard Biau and Erwan Scornet. “A random forest guided tour”. In: *Test* 25 (2016), pp. 197–227.

[45] Finn Gustafsson. “Comparing Random Forest, XGBoost and Neural Networks With Hyperparameter Optimization by Nested Cross-Validation”. In: ().

[46] Saeid Parvandeh et al. “Consensus features nested cross-validation”. In: *Bioinformatics* 36.10 (2020), pp. 3093–3098.

[47] Charles E Metz. “Basic principles of ROC analysis”. In: *Seminars in nuclear medicine*. Vol. 8. 4. Elsevier. 1978, pp. 283–298.

[48] Zachary C Lipton, Charles Elkan, and B Narayanaswamy. “Thresholding classifiers to maximize F1 score”. In: *stat* 1050 (2014), p. 14.

[49] Dimitri Boeckaerts et al. “Predicting bacteriophage hosts based on sequences of annotated receptor-binding proteins”. In: *Scientific reports* 11.1 (2021), p. 1467.

A Appendix

A.1 Probability distributions of each pipeline

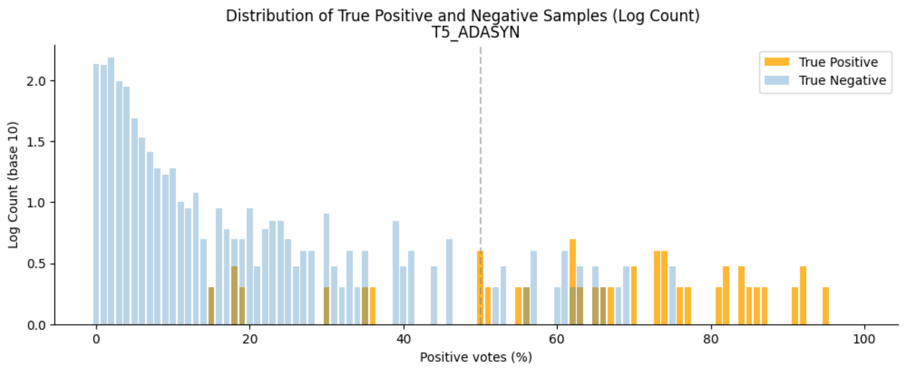


Figure A.1: Probability distribution of T5_ADASYN

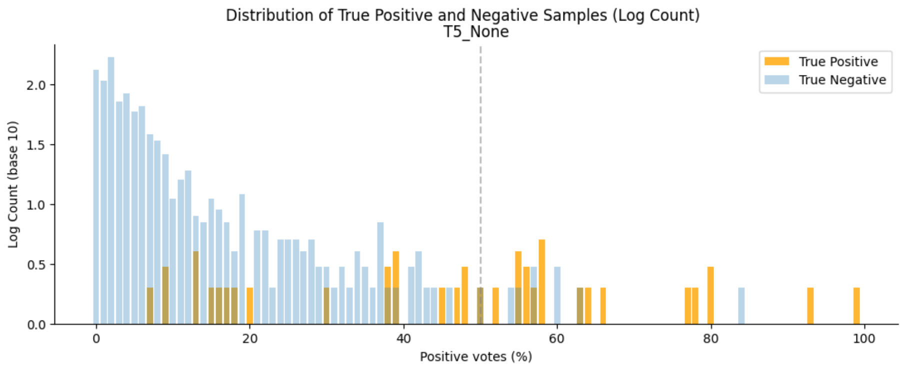


Figure A.2: Probability distribution of T5_None

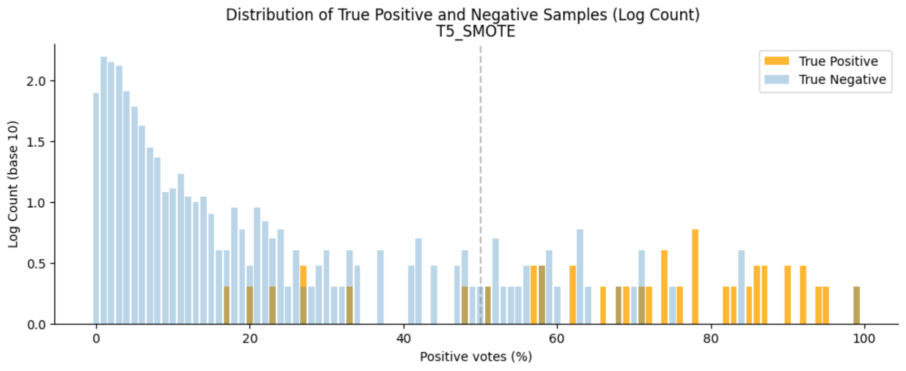


Figure A.3: Probability distribution of T5_SMOTE

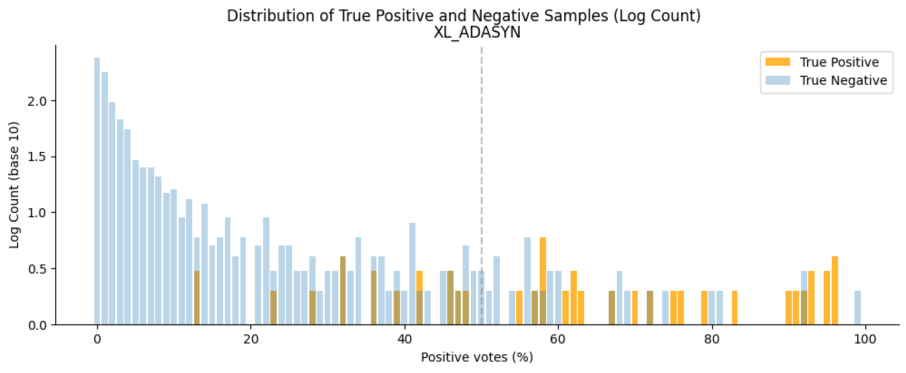


Figure A.4: Probability distribution of XL_ADASYN

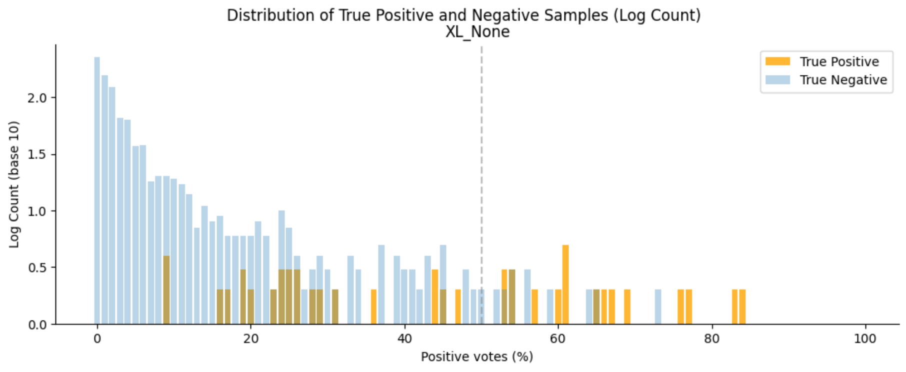


Figure A.5: Probability distribution of XL_None

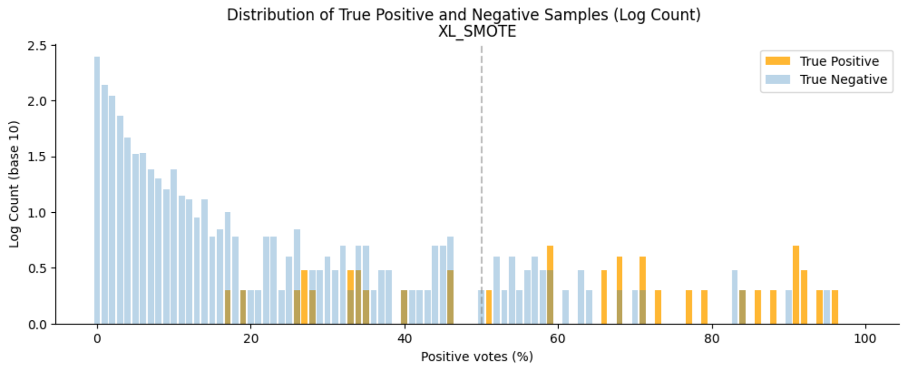


Figure A.6: Probability distribution of XL_SMOTE

A.2 Scoring and fitting times for each pipeline

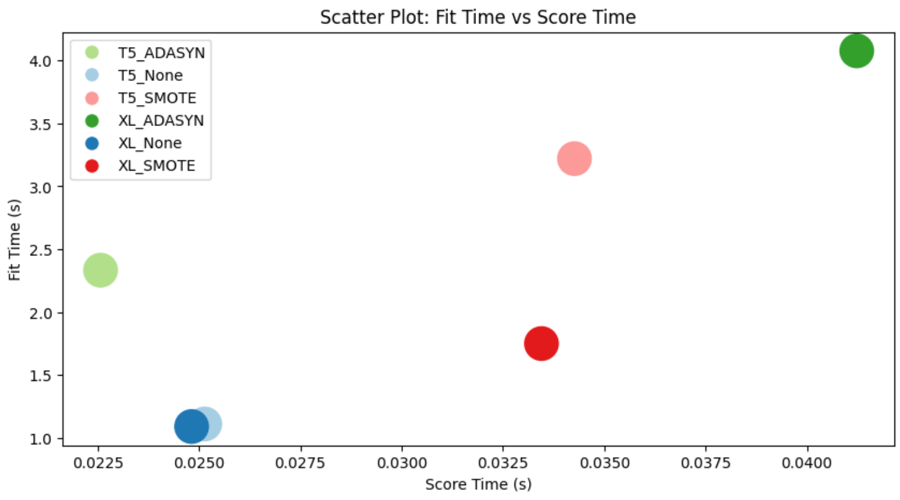


Figure A.7: Score and fit time of each pipeline