BRNO UNIVERSITY OF TECHNOLOGY
VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ

FACULTY OF INFORMATION TECHNOLOGY
FAKULTA INFORMAČNÍCH TECHNOLOGIÍ

DEPARTMENT OF INFORMATION SYSTEMS
ÚSTAV INFORMAČNÍCH SYSTÉMŮ

MACHINE LEARNING-BASED PREDICTION OF MUTATIONAL EFFECTS ON PROTEIN IMMUNOGENICITY

PREDIKCE VLIVU MUTACÍ NA IMUNOGENICITU PROTEINŮ ZALOŽENÁ NA STROJOVÉM UČENÍ

MASTER'S THESIS
DIPLOMOVÁ PRÁCE

AUTHOR AUTOR PRÁCE	DÁVID LACKO
SUPERVISOR VEDOUCÍ PRÁCE	Ing. MILOŠ MUSIL, Ph.D.
CONSULTANT KONZULTANT	STANISLAV MAZURENKO, Ph.D.

BRNO 2024

BRNO UNIVERSITY OF TECHNOLOGY
FACULTY OF INFORMATION TECHNOLOGY

Master's Thesis Assignment

Institut:	Department of Information Systems (DIFS)	157051
Student:	**Lacko Dávid, Bc.**	
Programme:	Information Technology and Artificial Intelligence	
Specialization:	Bioinformatics and Biocomputing	
Title:	**Machine Learning-based Prediction of Mutational Effects on Protein Immunogenicity**	
Category:	Biocomputing	
Academic year:	2023/24	

Assignment:

1. Review the literature on machine learning and protein engineering, protein immunogenicity, and immune response.
2. Review available tools for predicting immunogenicity.
3. Search for available datasets.
4. Collect data on protein immunogenic assays from the databases and develop an algorithm for identifying protein mutations and their effect on immunogenicity in the collected dataset.
5. Evaluate the performance of the available state-of-the-art tools in their ability to predict the effects of mutations on the newly generated dataset.
6. After consultation with the supervisor, design and implement a new predictor for this task.
7. Evaluate the results.

Literature:

- Clifford, J.N., Høie, M.H., Deleuran, S., Peters, B., Nielsen, M. and Marcatili, P., 2022. BepiPred-3.0: Improved B-cell epitope prediction using protein language models. Protein Science, 31(12), p.e4497.
- Sanchez-Trincado, J.L., Gomez-Perosanz, M. and Reche, P.A., 2017. Fundamentals and methods for T-and B-cell epitope prediction. Journal of immunology research.
- Vita, R., Mahajan, S., Overton, J.A., Dhanda, S.K., Martini, S., Cantrell, J.R., Wheeler, D.K., Sette, A. and Peters, B., 2019. The immune epitope database (IEDB): 2018 update. Nucleic acids research, 47(D1), pp.D339-D343.

Requirements for the semestral defence:
The first four points of the assignment.

Detailed formal requirements can be found at https://www.fit.vut.cz/study/theses/

Supervisor:	**Musil Miloš, Ing., Ph.D.**
Consultant:	Stas Mazurenko
Head of Department:	Kolář Dušan, doc. Dr. Ing.
Beginning of work:	1.11.2023
Submission deadline:	17.5.2024
Approval date:	30.10.2023

Abstract

The immune system is a vital part in human survival since it is responsible for protecting the body against pathogens. This ability stems from molecular mechanisms for the recognition of non-human proteins and molecules. While this system is critical for survival, it hampers the use of non-human proteins as biotherapeutics, many of which have already demonstrated significant potential in healthcare. To exploit this potential, it is vital that the immune system does not attack and inactivate the proteins. Therefore, it is often necessary to engineer these proteins to reduce the immunogenicity and avoid early detection by the immune system. To this end, scientists introduce mutations to a protein of interest to lower the response. Large-scale experimental validation of such mutations is typically unfeasible due to the enormous size of combinatorial space to explore. With the help of machine learning tools, this process can be accelerated and total development cost significantly reduced by scoring the mutations in silico first and experimentally validating only a subset of short-listed viable designs. However, the field of machine-learning-based tools for predicting such mutational effects is yet to be explored.

To address this challenge, we present a novel dataset focused on the effect of mutations on epitopes – protein regions that trigger the immune system response. The newly collected dataset contains epitopes, their single and double-point mutations, and the effect of these mutations on imunogenicity as labels. By leveraging this novel dataset and recent advances in large language models for protein engineering, we train a set of machine-learning-based models that are able to classify mutations based on their effect on immunogenicity, showing a significant improvement in performance over the baselines. Additionally, we investigate and present a way to separate the dataset into different train-test splits to minimize data leakage between these splits. This leads to a more robust real-world performance evaluation of the models trained on this data.

Abstrakt

Imunitný systém je dôležitou súčasťou prežitia človeka, pretože je zodpovedný za ochranu tela pred patogénmi. Táto schopnosť vyplýva z molekulárnych mechanizmov rozpoznávania cudzorodých bielkovín a molekúl. Hoci je imunitný systém rozhodujúci pre prežitie, bráni využívaniu proteínov pochádzajúcich z iných organizmov ako bioterapeutík, z ktorých mnohé už preukázali významný potenciál v zdravotníctve. Na využitie tohto potenciálu je nevyhnutné, aby imunitný systém tieto proteíny nenapadol a nedeaktivoval. Preto je často potrebné tieto proteíny upraviť tak, aby sa znížila ich imunogénnosť a zabránilo sa ich detekcii imunitným systémom. Na tento účel vedci zavádzajú mutácie do proteínu, ktorý je predmetom záujmu, aby znížili imunitnú odpoveď. Rozsiahle experimentálne overovanie takýchto mutácií je zvyčajne neuskutočniteľné vzhľadom na obrovskú veľkosť kombinatorického priestoru, ktorý treba preskúmať. Pomocou nástrojov strojového učenia možno tento proces urýchliť a výrazne znížiť celkové náklady na vývoj tým, že sa mutácie najprv vyhodnotia in silico a experimentálne sa overí len podmnožina sľubných návrhov z užšieho výberu. Oblasť nástrojov založených na strojovom učení na predpovedanie takýchto mutačných účinkov však ešte nie je preskúmaná.

Na vyriešenie tejto výzvy predstavujeme nový súbor dát zameraný na vplyv mutácií na epitopy - oblasti bielkovín, ktoré spúšťajú reakciu imunitného systému. Novo zhromaždený súbor dát obsahuje epitopy, ich jednobodové a dvojbodové mutácie a vplyv týchto mutácií na imunogénnosť. Využitím tohto nového súboru a nedávnych pokrokov v oblasti veľkých jazykových modelov pre proteínové inžinierstvo sme natrénovali súbor modelov založených na strojovom učení, ktoré sú schopné klasifikovať mutácie na základe ich vplyvu na imunogenicitu, pričom vykazujú výrazné zlepšenie výkonu oproti existujúcim a základným modelom. Okrem toho prezentujeme spôsob rozdelenia súboru dát na rôzne tréningovo-testovacie rozdelenia s cieľom minimalizovať prienik údajov medzi týmito rozdeleniami. To vedie k spoľahlivejšiemu ohodnoteniu reálnej výkonnosti modelov natrénovaných na týchto údajoch.

Keywords
machine learning, immunoinformatics, protein engineering, immunogenicity prediction

Kľúčové slová
strojové učenie, imunoinformatika, proteínové inžinierstvo, predikcia immunogenicity

Reference
LACKO, Dávid. *Machine Learning-based Prediction of Mutational Effects on Protein Immunogenicity.* Brno, 2024. Master’s thesis. Brno University of Technology, Faculty of Information Technology. Supervisor Ing. Miloš Musil, Ph.D.

Rozšírený abstrakt

Ľudský imunitný systém zohráva rozhodujúcu úlohu pri obrane nášho tela pred napadnutím patogénmi. Kľúčovým prvkom tohto obranného mechanizmu je rozpoznávanie cudzích molekúl, vrátane proteínov, špecializovanými imunitnými bunkami nazývanými lymfocyty. Toto rozpoznávanie prebieha na základe špecifických oblastí antigénov nazývaných epitopy. Epitopy možno vo všeobecnosti rozdeliť na dva typy: lineárne epitopy, ktoré pozostávajú zo súvislej sekvencie aminokyselín, a štrukturálne epitopy, ktoré sú trojrozmerné konformácie tvorené nespojitými úsekmi proteínovej sekvencie.

Vývoj nových liečiv na báze bielkovín často zahŕňa využitie bielkovinových predlôh z rôznych organizmov. Tieto, pre ľudské telo cudzorodé bielkoviny, však môžu pri vstupe do organizmu vyvolať nežiadúce imunitné reakcie, najmä ak je zdrojový organizmus známy patogén. Preto je kľúčovým krokom pri návrhu liečiv na báze bielkovín ich úprava, s cieľom minimalizovať ich imunogénnosť, teda schopnosť vyvolať imunitnú odpoveď. Tradične sa úprava imunogenicity bielkovín vykonáva pomocou pracných a finančne náročných experimentálnych techník na identifikáciu a modifikáciu imunogénnych epitopov. Strojové učenie ponúka nástroje na zefektívnenie tohto procesu. Algoritmy strojového učenia dokážu analyzovať veľké súbory dát známych epitópov a sekvencií bielkovín s cieľom identifikovať vzorce, ktoré možno použiť na predpovedanie imunogenity nových sekvencií.

Napriek tomu, že identifikácii epitópov pomocou strojového učenia je venovaných viacero nástrojov a článkov, tak doteraz nebol navrhnutý žiadny prediktor, ktorý by vyhodnocoval účinky mutácií na imunogenitu. Táto práca si kladie za cieľ vyplniť túto medzeru a vyvinúť prediktor založený na strojovom učení, ktorý možno použiť na posúdenie vplyvu mutácií na imunogenitu bielkovín.

Hlavnou výzvou pri vytváraní takéhoto modelu je získanie kvalitných dát na trénovanie a testovanie. V tejto práci je navrhnutý nový robustný proces získavania a spracovania dát. Využívame existujúce databázy epitopov na extrakciu relevantných informácií o nich a o ich zdrojových bielkovinových sekvenciách. Tieto údaje sa následne spracovávajú s cieľom získať informácie o mutáciách, a zároveň sa zabezpečuje ich konzistentnosť. Ďalším aspektom nášho prístupu je vytvorenie kvalitného rozdelenia dát do trénovacích, testovacích a validačných dátových sád, za účelom minimalizovať presak informácii medzi nimi, a tým pádom zaručiť kvalitné ohodnotenie modelov.

Z databázy IEDB sa nám celkovo podarilo vyťažiť 8742 záznamov o mutáciach epitopov, a z toho je 1584 označených ako mutácii znižujúcich imunogenitu. Databáza IEDB sama o sebe neobsahuje mutačné záznamy, ale vďaka porovnávaniu a prelínaniu existujúcich záznamov bolo možné vyťažiť informácie o mutáciách. Zároveň sme dáta rozdelili do 3 nezávislých častí na trénovanie, testovnaie a validáciu.

Cieľom je vytvoriť model založený na strojovom učení, ktorý pomôže pri vývoji nových liečiv tým, že umožní výskumníkom predpovedať, ako mutácie, ktoré zaviedli do epitopov, ovplyvnia imunogenicitu bielkoviny. Tento model môže byť užitočný hlavne keď predloha liečiva pochádza z iného organizmu. Pretože je nutné bielkovinovú predlohu upraviť tak, aby nedráždila imunitný systém človeka. S pomocou prezentovaného modelu môžu výskumníci strategicky a s nižšími nákladmi testovať rôzne mutácie epitopov s cieľom minimalizovať imunogenitu.

Aby sme overili spoľahlivosť existujúcich nástrojov na mutačných dátach, tak sme už existujúce nástroje na identifikáciu epitopov adaptovali na mutačné dáta a ohodnotili na novej dátovej sade. Zistili sme, že na výsledky týchto modelov sa nedá spoľahnúť pri vyhodnocovaní vplyvu mutácii na epitopy. Preto prezentujeme nové modely, ktoré sa zameriavajú na túto úlohu a dosahujú oveľa vyššiu presnosť. Na ich vytvorenie sme použili predtrénovaný model ESM2 v kombinácii s algoritmami strojového učenia. Najlepší model dosahuje presnosť 0.73, čo je v súčasnosti state-of-the-art.

Machine Learning-based Prediction of Mutational Effects on Protein Immunogenicity

Declaration

I hereby declare that this Diploma thesis was prepared as an original work by the author under the supervision of Ing. Miloš Musil, PhD. and Stanislav Mazurenko, PhD. I have listed all the literary sources, publications and other sources, which were used during the preparation of this thesis.

....
Dávid Lacko
May 16, 2024

Acknowledgements

I would like to thank my supervisor Miloš Musil, but especially my consultant Stanislav Mazurenko for giving his time to guide me and for the weekly consultations. I am also greatful to the Loschmidt laboratories team for including me in their projects and helping with the thesis.

Contents

Chapter 1

Introduction

The human immune system plays a critical role in defending our bodies against invading pathogens. A key element of this defense mechanism is the recognition of foreign molecules, including proteins, by specialized immune cells called lymphocytes. This recognition relies on specific regions of the foreign protein called epitopes. Epitopes can be broadly classified into two types: linear epitopes, consisting of a continuous sequence of amino acids, and structural epitopes, which are three-dimensional conformations formed by discontinuous segments of the protein sequence.

The development of novel protein-based drugs often involves utilizing protein templates from different organisms. However, these foreign proteins can trigger unwanted immune responses when introduced into the human body, especially if the source organism of these templates is a known pathogen. Therefore, a crucial step in protein-based drug design is to engineer these proteins to minimize their immunogenicity, the ability to induce an immune response.

Traditionally, engineering protein immunogenicity has involved laborious and expensive experimental techniques to identify and modify immunogenic epitopes. Machine learning (ML) offers a promising approach to streamline this process. ML algorithms can analyze large datasets of known epitopes and protein sequences to identify patterns that can be used to predict the immunogenicity of novel sequences. Despite the progress in epitope identification with ML, no predictor of mutational effects on immunogenicity has ever been proposed. This thesis aims to fill this gap and develop a machine learning-based predictor that can be used to assess the impact of mutations on protein immunogenicity.

A central challenge in building such a predictor lies in acquiring high-quality data for training and evaluation. This thesis addresses this challenge by proposing a robust data mining and processing pipeline. We leverage existing databases to extract relevant information on epitopes and protein sequences. This data is then meticulously processed to mine mutational information, while ensuring accuracy and consistency. A critical aspect of our approach is the creation of well-defined training, test, and validation data splits to minimize data leakage and ensure the generalizability of our model.

Our goal is to create a machine learning-based tool that aids in the design of novel drugs by enabling researchers to predict how mutations introduced into identified epitopes will affect the protein's immunogenicity. This tool can be particularly valuable when the drug template originates from a different organism, requiring modifications to evade the human immune system. By utilizing our predictor, researchers can strategically mutate epitopes to minimize immunogenicity while potentially preserving the efficacy of the protein-based drug.

This thesis is structured as follows. Chapter 2 provides an introduction to proteins, protein engineering, and biopharmaceuticals. Chapter 3 highlights current knowledge about the immune system, more specifically about the adaptive immune system, and describes the concept of immunogenicity. Chapter 4 contains a review of machine learning, existing methods for epitope prediction, and machine learning approaches used in immunogenicity prediction. Chapter 5 presents the source data and development of the pipeline to create the novel dataset. Chapter 6 describes the development of our proposed machine learning-based predictor for mutational effects on protein immunogenicity. Chapter 7 contains results, performance metrics and comparison of the existing and newly proposed models. Finally, Chapter 8 concludes the thesis by summarizing the key contributions of this work.

Chapter 2

Proteins

Proteins, ubiquitous building blocks of life, facilitate a multitude of vital functions within organisms, from catalyzing chemical reactions to manipulating and repairing DNA. They provide structural stability to cells, transport of metabolites within and outside cells, help fight pathogens, and they are also at the core of all active movement of organisms [2]. From a chemical perspective, proteins are macromolecules that comprise long chains of amino acids. These chains are usually also specifically arranged in the 3D space to be able to perform their function. When analyzing proteins in silico, there are multiple representations, from a string of letters representing amino acids to a set of 3D atom coordinates.

The basic units of proteins are amino acids. They have a unique chemical structure that allows them to form peptide bonds. On the one end, they have a carboxylate functional group, and on the other end, an amino functional group. Under certain conditions, these two groups can join together and thus form a chain. In addition, each amino acid also has a side chain, which differentiates one amino acid from the other. This side chain determines the properties of an amino acid and helps to form a 3D structure and binding sites of proteins. Although more than 500 different amino acids are known today, only 20 of them are encoded in the standard genetic code of all living organisms, and two more are sometimes incorporated by special translation mechanisms (Table 2.1).

Table 2.1: The most common amino acids occurring in proteins and the corresponding one-letter notation typically used in the field of protein engineering to represent protein sequences.

Amino acid	letter	Amino acid	letter	Amino acid	letter
Alanine	A	Glycine	G	Proline	P
Arginine	R	Histidine	H	Serine	S
Asparagine	N	Isoleucine	I	Threonine	T
Aspartate	D	Leucine	L	Tryptophan	W
Cysteine	C	Lysine	K	Tyrosine	Y
Glutamine	Q	Methionine	M	Valine	V
Glutamate	E	Phenylalanine	F		

The AAindex[1] database serves as a comprehensive resource for studying the properties of the 20 standard amino acids. There are 566 chemical, physical, and statistical descriptors of these amino acids. Additionally, the database includes mutation matrices and pair-wise contact potentials. All of these properties can be used to study the roles of amino acids within proteins and to engineer features for machine learning applications.

2.1 Protein synthesis

In organisms, proteins are produced by a complex process that starts with the genetic information of the organism and ends with the functional protein. First, the genes stored in the DNA that encode a certain protein are copied by a process called transcription. The part of the DNA that contains the gene is copied to mRNA. The ability of the DNA to copy arises from the complementarity of the nucleic acid bases, which constitute the DNA.

After the mRNA is synthesized, it can be translated into a peptic chain, which is done by ribosomes. They can translate 3 mRNA bases that are next to each other into a singular amino acid and catalyze the formation of peptic bonds between neighboring amino acids. Repeating this action over and over leads to the elongation of the peptide chain. The order in which the amino acids are bound to each other forms the primary structure of the protein. When enough amino acids with favorable properties are close to each other, they form the so-called secondary structure elements in the 3D space: an α-helix, a β-sheet, or a turn. When there are not enough favorable amino acids, it is also possible that they do not take a 3D shape and stay in the form of a disordered chain. Finally, after the entire protein has been synthesized, it can take its final shape in a 3D space, which is called the tertiary structure. Here, all the secondary shapes remain formed, but they interact with each other and with other parts of the chain to take their final positions. It is also worth noting that protein molecules are dynamic in nature, the property often critical for their function, and their tertiary structure is, in fact, a changing conformation, which can be regarded as a sample from the underlying energy distribution based on the protein environment and state.

In some cases, the protein cannot fold on its own and requires the help of other proteins called chaperones, which are specialized for this task. The correct function of the protein is greatly impacted by this structure. For instance, a protein in a denatured state still typically has the same primary and partially secondary structure, but it usually loses its function. The denatured state of a protein is defined as a state in which the protein loses its tertiary structure due to some external circumstances, such as a change in acidity, temperature, pressure, or other factors.

Protein synthesis differs between prokaryotes and eukaryotes. Since prokaryotes do not have a nucleus, DNA is freely present in plasma, and both transcription and translation processes can occur at the same time. The eukaryotes have nuclei that contain the DNA but not the translation mechanism, so the mRNA is first transcribed, then optionally processed, then it has to leave the nucleus to get translated.

2.2 Taking advantage of proteins

Proteins are essential for living organisms, but humans have also found ways to utilize them to their advantage, e.g., in chemical, medical, pharmaceutical, and biotechnological industries, for example, by overexpressing required proteins using cell cultures. The most

[1] https://www.genome.jp/aaindex/

prominent use is the production of alcohol and dough with yeast, which has been around for millennia. One of the first out-of-cell uses of proteins was for laundry detergent in 1913, but the enzyme was not stable enough. Only in the 1960s, a new enzyme was discovered that became an important additive to detergents for home and industry use [23]. Another example of the use of proteins in industry is the use of new lubricants for machines [31]. In the medical field, insulin, a hormone responsible for the absorption of sugar from the blood, has profoundly improved the life of people with diabetes. Recently, the use of monoclonal antibodies has become more and more popular to restore or modify the immune system's response to pathogens and cancer cells. Some enzymes are also used as biosensors to detect the presence of different substrates, as well as solutions to many modern environmental challenges, such as carbon capture or plastic degradation [38].

For proteins to fulfill their function, they have to bind to other molecules, which are called ligands. **Enzymes** are proteins that catalyze chemical transformations, increasing the rate of reactions by lowering the energy barrier required for the reaction to happen. After the reaction is complete, most enzymes go back to their native form, making them ready for the next reaction cycle. In this way, organisms can produce essential chemicals and regulate themselves.

Proteins have a wide variety of functions, but with that also come different environmental requirements. Some proteins are produced to never leave the cell, others are signaling molecules that are released into the environment, and some enzymes facilitate reactions outside the cell. The conditions outside and inside the cell might be entirely different. The inside of the cell is highly regulated and controlled, whereas the outside may change rapidly. All of this impacts the evolution trajectory each protein has undergone. Some organisms can survive at relatively high temperatures, even above 80 ℃, while some human proteins start to degrade already at 40 ℃. This particular property is called protein stability and is one of the basic attributes of proteins. It is often quantified by the temperature at which half of the protein sample is denatured (the melting temperature).

Another important property is pH stability, especially for the industrial use of proteins, where the pH might be much higher or lower than the pH in their natural environment. This was also the reason why the initial use of proteins in detergents failed in 1913 when the researchers used trypsin in combination with soap, which changed the pH of the solution and the trypsin became much less active.

This brings us to another critical property, particularly for enzymes, the activity. It describes how well the protein facilitates the reaction. Some enzymes might have higher activity but are more sensitive to the environment, while others might have lower activity but can facilitate many reactions. This property is also interesting from the point of industrial use, where we usually want the fastest reaction possible, but the particular enzyme evolved with lower activity because the reaction it facilitates had to be regulated inside the cell.

For example, consider lactate dehydrogenase, the enzyme that catalyzes the reaction of lactate to pyruvate and back, which is a basic reaction in the anaerobic energy pathway as lactate is a molecule that builds up in muscles during high-strain activities due to the lack of oxygen [17]. The enzyme could potentially have evolved to have higher activity, but if it had, it would process pyruvate too fast, and create too much lactate, which could lead to acidosis. However, in industrial conditions, it is used to produce all kinds of fermentation products and lactic acid, where a higher activity would be beneficial.

2.3 Protein engineering

The discipline of studying individual proteins and the ways to modify them is called protein engineering. Scientists in this field usually first study the sequence, structure, and behavior of proteins under natural and artificial conditions. They then try to modify existing proteins to have better properties, such as higher melting temperatures or better activity. Another field of interest is to study a certain reaction and a family of enzymes that catalyze this reaction to better understand and improve it.

A newly emerging field is de novo protein design, where scientists do not modify the existing proteins but instead attempt to design completely new proteins from scratch [67]. This was previously not possible because the determination of the structure of proteins is a challenging task, even for natural proteins. However, today, with the emergence of complex AI-based tools, such as AlphaFold [28], a new generation of protein engineering tools is emerging that are capable of using the structure of a protein that was not determined experimentally before.

2.4 Biopharmaceuticals

Biopharmaceuticals are pharmaceutical drugs that are extracted or semisynthesized from biological sources. They can be categorized into multiple classes based on their composition and use:

- Protein-based therapeutics
- Nucleic acid-based therapeutics
- Gene therapies
- Vaccines

There are a total of 566 of approved therapeutics in these categories in the U.S. and E.U. at the time of writing. If we subtract the vaccines from the list because they fall into prevention rather than treatment and have different principle of function, then there are 378 therapeutics in total, 349 of them being protein-based therapeutics [33].

Most of the drugs and therapeutics used historically are in the category of small molecules. Small molecules do not fall under the biopharmaceuticals and have low molecular weight of ($\leq$ 1000 Da). They are small, mostly organic, compounds that regulate biological processes. Since they are used in drugs, they have to be approved by regulatory entity. When searching for novel treatments these molecules are screened first, since it is easier to approve a new drug with already approved chemical compounds. Currently, there are 1973 of them approved in the U.S. and E.U.

Some protein-based therapeutics are peptides, which have a molecular weight of $500 - 5000$ Da. These include insulin, oxytocin, vasopresin, and more. From 2000 to 2022, a total of 33 non-insulin peptide drugs have been approved worldwide, for various use including treatment of advanced prostate cancer, multiple myeloma, osteoporosis, etc. Most of them are peptides mimicking human peptides, but some of them are derived from other organisms.

The rest of the protein-based therapeutics are proteins with molecular weights above 5000 Da, such as monoclonal antibodies, thrombolytics (alteplase, tenecteplase), recombinant clotting factors (Turoctocog alfa), hormones, growth factors, and vaccines. The source organism of many of these proteins are humans, meaning they are natural to the human body and their codes are part of our genome. They are administered because there might be a deficiency, mutation, or other problem, and the body is not able to synthesize them on its own. In other cases, e.g., with ischemic stroke, alteplase, a thrombolytic drug capable of dissolving blood clots, can be administered. In this case, alteplase is naturally occurring in the blood stream, but in insufficient amounts to dissolve the clot in time.

Few of these proteins come from other organisms, such as chymopapain [15], which comes from *Carica papaya* and is a cysteine protease. These proteins, however, are recognized as a threat by the immune system and can be attacked and degraded. For chymopapain, it is not a problem since it is administered acutely and using injection into the affected area to take effect. However, there are enzymes that need to be administered to the bloodstream to be distributed throughout the body and are thus more sensitive to immune responses. One example is Staphylokinase [11] [22], a thrombolytic drug that could be used to treat acute ischemic stroke. It comes from a bacteria called Staphylococcus aureus. Compared to the approved alternative alteplase, it shows a more specific and faster response, while causing less internal bleeding after administration.

Chapter 3

Immune system

The immune system is one of the most complex body systems and has multiple parts that have to cooperate to defend the body efficiently. It consists of a distributed network of cells, organs, proteins, and chemicals. The main goal of this system is to protect against foreign and internal threats. It protects the host from other organisms, toxins, parasites, and viruses, generally called pathogens. In multicellular organisms, its task is also to eliminate mutated cells that can cause cancer. Almost all living organisms have some kind of immune system, ranging from simpler systems in bacteria to the most complex in mammals. The system can be divided into two subsystems, innate and adaptive [1].

3.1 Innate immune system

This system is older and not as specific as the adaptive one. It is usually the first line of defense against microbial and virus pathogens. Its main distinguishing feature is that it does not learn from previous encounters. Therefore, it responds in the same way if the same pathogen infects the host for the second time. Its main reactions are to induce inflammation at the sites of infection and to defend against viruses by eliminating virus-infected cells.

This immune subsystem can recognize pathogens due to pathogen-associated molecular patterns (PAMPs). These molecules are inherently associated with certain types of pathogens, such as peptidoglycans that are part of the cellular wall of some bacteria. Other examples include flagellin, which is a protein that forms the bacterial flagellum, or double-stranded RNA, which is typical for some viruses. All PAMPs are typical for certain pathogens, and these pathogens cannot easily mutate these patterns because they are essential for their survival, making their PAMPs effective and reliable targets to recognize pathogens. It is estimated that for humans around 1000 different PAMPs exist.

Additionally, the innate immune system also has receptors to recognize damage-associated molecular patterns. These are present when cells break down and are detected to mobilize innate immunity to remove debris and start the repair process.

Some examples of cells that are part of this subsystem are:

- neutrophils – the most abundant leukocytes in the blood, they are the first to respond to a bacterial or fungal infection. They are phagocytes, which means that they can engulf pathogens and digest them inside.

- macrophages – their precursors are present in the blood, and after they are activated, they become macrophages. Their function is similar to that of neutrophils, they phagocyte the pathogens and release and regulate inflammation, but they have a much longer life span.

- dendritic cells – they are present in tissues that are part of the microbial response, such as lymphatic nodes. They work closely with the adaptive immune system by presenting antigens from processed pathogens.
- mast cells, natural killer cells, etc.

In addition, part of this immune system is also the barrier provided by the skin and mucosal tissues present, for example, in the nose and mouth.

It is important to note that this subsystem combats either bacteria directly or viruses indirectly, since it targets cells infected by viruses. The innate immune system is incapable of producing antibodies, and thus it cannot affect proteins floating freely in the environment. Moreover, after the cells that comprise this system differentiate to their final state, they all have the same receptors, which is different in the adaptive immune system.

3.2 Adaptive immune system

The adaptive immune system is phylogenetically younger but more powerful and present only in vertebrates. The complexity of the adaptive immune system is based on its ability to learn, which creates large variations in the population and even within one organism. This system responds slower than the innate system because the cells that can respond first need to proliferate after encountering the threat. Its main characteristic is that it can learn, so the second encounter with the same pathogen triggers a much faster and stronger response. It is essential in the fight against pathogenic organisms that can adapt to avoid the innate immune system, e.g., viruses. These are usually infectious diseases that are harmful to humans.

The innate system is capable of recognizing general patterns shared by entire families of microbes. In contrast, the adaptive immune system can adapt and recognize proteins or other molecules that are specific to a particular organism. In this way, the latter can recognize a wide variety of molecules, regardless of whether they are harmful or not, and these molecules are called **antigens** in this context. The specific parts of antigens that the adaptive immune system can recognize are called **epitopes** [20]. An **epitope** is a specific part of the molecule that directly binds to antibodies or receptors, and this binding causes the immune system to identify the given molecule as pathogenic. In protein targets, it is either a short subsequence (linear epitopes) or a set of amino acids that are in close vicinity in the 3D structure of the protein (structural epitopes).

The adaptive immune system has two main parts. The first part is the cells that digest and process pathogens and present processed antigens. These include dendritic cells, macrophages, and mast cells, commonly called antigen-presenting cells (APCs), which were mentioned as part of the innate immune system, but are also part of the adaptive immune system. The second part is T cells and B cells, called lymphocytes, which carry out adaptive immune system responses based on the antigens presented by the APCs. Lymphocytes can recognize specific pathogens thanks to receptors on their surfaces. These receptors are specific for each cell line, making them very diverse.

3.2.1 T cells

T cells are created in the bone marrow and then travel to the thymus, where they undergo a mutation process and develop their specific receptors through a process called the V(D)J recombination process, where DNA is edited and spliced. In this process, the genes that encode the receptors are mutated randomly but not chaotically to form a novel random receptor.

T cell receptors [40] react to antigens presented by class I or class II MHC proteins. These are specialized proteins that bind to antigen proteins and present them on the surfaces of cells. T cells have two types of receptors, the first receptor reacts to the MHC protein, and the second receptor reacts to the antigen presented on the MHC protein. This mechanism ensures that T cells will not be activated anytime the second receptor binds, but only if it binds to the antigen presented by the MHC-presenting cell, which is confirmed by the first receptor.

After T cells develop their receptors and express them on their surface, they undergo a positive and negative selections. The positive selection keeps T cells that developed functional first receptors that correctly recognize MHC proteins, manifesting in low or moderate affinity for the antigens presented with MHC. This is possible because of cells in the thymus that present, using the MHC, a wide variety of different proteins and molecules from all over the body (self-antigens). The negative selection then filters out T cells that developed receptors that would bind strongly to self-antigens, i.e., they would recognize the body cells as foreign and attack them. This manifests itself in a very high affinity towards the self-antigens presented with MHC because both receptors bind. Both of these processes help regulate T cells and reduce the probability of autoimmune diseases.

After the selection of T cells, they are released into the blood and plasma, where they circulate until they find the antigen they can recognize. If they do, they start to divide and proliferate, creating copies of themselves while maintaining the same receptor. They also promote inflammation and excrete chemicals that should kill pathogens. Other copies, called helper T cells, help activate B cells. T cells are also responsible for the elimination of cancerous cells.

3.2.2 B cells

The B cells are the only cells capable of producing antibodies. **Antibodies** are proteins composed of 4 polypeptide chains, 2 are identical light chains and 2 are identical heavy chains, all joined by disulfide bridges. Generally, antibodies have a Y shape, but some variants of antibodies have multiple Y-shaped units joined together in a circle, forming a star pattern. Both light and heavy chains contain a variable region, which is responsible for a great variety of antibodies because, during cell development, it is heavily mutated by the V(D)J process [3]. This process is theoretically capable of making more than 10^{11} distinct variable regions in antibodies. The purpose of antibodies is to bind to antigens and either disable them by the binding or mark them for destruction by other cells of the immune system. Some antibodies are excreted from B cells, while other light and heavy chains make part of B cell receptors. The antibodies are especially effective against viruses because they can disable them before they infect the cell by binding to the proteins responsible for allowing the virus to enter the cell. The place where they bind to the antigen is called the epitope.

The main difference between B-cell and T-cell epitopes is their structure. T-cell epitopes are mostly linear, meaning that T-cell receptors recognize peptides based on their amino acid sequences. This is caused by the binding of MHC and antigen preprocessing. In contrast, B-cell antibodies and receptors bind directly to antigens, which can be fully assembled proteins. This causes the distribution of B-cell epitopes to be 90% structural and 10% linear.

B cells are also created in the bone marrow, but they undergo differentiation there and travel to the spleen to complete their maturation. They have, similarly to T cells, the V(D)J recombination process that results in a wide variety of antibodies. They are also selected, but differently compared to T cells. Their positive selection is focused only on filtering the correct antibodies and receptors, which means that if something goes wrong during DNA recombination and the receptors are malformed, those cells undergo apoptosis. Their negative selection is also different. If the B cell binds strongly to some self-antigen, then it can undergo apoptosis, or it can undergo V(D)J recombination again and change its receptor. This selection is not as thorough as in T cells and is done only against self-antigens that are ubiquitous in the body, such as blood proteins or membrane molecules present in all cells. Only B cells that would react with most body cells are removed, while those that might bind to proteins specific to a certain part of the body are kept. Mature B cells usually reside in lymph nodes and scan their environment, dendritic cells, and other antigen-presenting cells for pathogens. In addition, they are also capable of expressing the MHC protein and presenting antigens to T cells. As mentioned above, their antigens are more diverse and capable of recognizing proteins, lipids, nucleic acids, polysaccharides, and other types of molecules.

B cells rarely respond to protein antigens on their own, even if they have the corresponding receptor. This is because they need the help of T cells to confirm that it is indeed pathogenic and not just a body protein they have not seen before. If they encounter a protein to which their receptor strongly binds, they phagocytize this antigen, split it up, present it using the MHC complex, and wait for confirmation from the T cell that this is indeed a foreign protein [45]. If the antigen is not a protein, they can respond on their own. This form of regulation is one of the reasons why B cells do not have to undergo such a strong selection compared to T cells.

When B cells are activated, they start to proliferate and enter the bloodstream, where they begin to mass-produce antibodies to combat the antigen by which they were activated. Furthermore, these replicated B cells can further improve the antibody by small mutations to have a better affinity towards this antigen.

3.3 Immunogenicity

Immunogenicity is the term used to describe how strongly the immune system responds to a given substance or organism. This property is important for vaccine and drug design and development as it determines drug effectiveness [12]. Therefore, tools that can help affect the immunogenicity of a specific protein are in high demand as they can save time and resources spent during the drug development cycles.

The most studied immunogenicity effects are by the adaptive immune system, mainly due to the development of vaccines and new antibodies to battle new viruses or other pathogens and their variations. It is necessary to say that each individual has their unique set of antibodies that arise from exposure to different environmental conditions and pathogens in the past. By exposing the body to a new pathogen, the adaptive immune system is usually able to create new antibodies for this pathogen – the main mechanism behind vaccination.

When a new vaccine is developed, it is usually based on the antigen that would bind strongly to antibodies so that the lymphocytes would get activated and start replicating and producing antibodies. Thus, if the body encounters the real pathogen, the lymphocytes would have already multiplied and antibodies are abundant in the blood to fight the pathogen.

With protein-based therapeutics that are administered intravenously, the exact opposite effect is necessary: one wants to design the drug in such a way that the immune system will not attack it. This is typically not a problem for T cells because they will not destroy the protein. However, antibodies can bind to the protein and thus disable it. That is also a reason why most protein-based therapeutics are taken from human proteins: this approach eliminates the antibody problem, since the immune system has already applied the negative selection so as not to attack native proteins. In contrast, when a protein from another organism is considered, there is a high chance that the immune system will attack it because that is what the immune system has evolved to do.

This effect is different in the drugs administered acutely, e.g., once or twice in life, or the drugs that are administered periodically. An example of the first type of drug is a thrombolytic agent, which is administered when a person suffers a stroke [10]. An example of the second type is a protein-based drug, insulin, administered daily [56]. The first group of drugs is expected to work for an individual at least for the first time because the immune system has not seen the protein before, and if the amount is not too large, there is a chance that the immune system will not be triggered by the drug. Still, this might be a problematic neglect in the design of a drug, especially if it is designed to be administered to a sick person, since if their immune system gets triggered, there is a possibility of fever induced by the presence of the drug in the bloodstream. That is why the immunogenicity of protein-based drugs is so important, but also why drugs based on proteins from other organisms are not very common.

To explore the great potential of proteins from other organisms that can be used as therapeutics, it is thus necessary to decrease or remove immunogenicity, i.e, to disable the epitopes. There are multiple strategies to accomplish this goal, such as shielding methods or epitope removal [69]. Shielding methods include PEGylation, methylation, glycosylation and more. They always include attachment of another molecule to the protein to deny access of receptors or antibodies to the binding site. One of the strategies of epitope removal is domain removal, where a part of the protein is removed that contains or forms an epitope. Although, this strategy is relatively simple, it is not always possible to delete part of the protein and still keep its function, as the protein might not fold or the epitope might be a part of the catalytic pocket if the given protein is an enzyme.

Developing effective drugs often necessitates a more in-depth understanding of the target protein's epitopes, both structural and linear. Ideally, these epitopes can be strategically disabled or their immunogenicity reduced. However, studying epitopes experimentally proves to be a significant hurdle. Structural epitope analysis requires obtaining the protein's structure bound to an antibody, a complex and expensive undertaking. Conversely, linear epitope investigation involves fragmenting the protein into smaller peptides and testing them individually against various antibodies – a laborious and time-consuming process. Considering that protein development might necessitate multiple rounds of such experiments, the cost becomes a significant barrier. This highlights the critical need for efficient in silico tools to aid in epitope research. Such tools, potentially based on machine learning, hold immense promise for accelerating drug development and furthering our understanding of the immune system.

Chapter 4

Machine learning

Machine learning (ML) [57] [42] is a subfield of computer science concerned with the development of algorithms that can automatically identify patterns and relationships within data. These algorithms are trained on a representative subset of data, known as the training set or training data, enabling them to learn and make predictions on new, unseen data that has similar form to the training data. Machine learning has been utilized in many applications for more than 15 years, such as customized search results, suggestions on social networks, movie suggestions on online streaming platforms, spam filtering and much more. This was more behind-the-scenes usage, and the users might not have been aware of it. However, ML is becoming more and more visible with the emergence of large language models from companies like OpenAI [49] and Google [63] and their use as virtual AI assistants.

The machine learning techniques are usually divided into multiple categories:

1. **Supervised learning** – the training data includes input features and expected outputs. The algorithm looks at the inputs and tries to determine how to derive the outputs. This includes tasks such as classification and regression.

2. **Unsupervised learning** – the training data includes only input features. The algorithm tries to uncover the structure or groupings within the data. The typical tasks are dimensionality reduction and clustering.

3. **Reinforcement learning** – the training involves an interactive learning process between an agent and its environment. The agent learns by trial and error while being rewarded for desirable actions and penalties for not-desirable ones.

Machine learning is revolutionary because of its data-driven approach, without the need for analytical solution. In the field of bioinformatics, ML algorithms are employed in a multitude of ways, such as analyzing biological sequences, predicting protein structures and functions, designing novel drugs, antibodies and more. Since biological systems are so complex, machine learning helps to uncover previously hidden relations and process the increasing amounts of biological data.

The usual steps when solving problems using machine learning are as follows:

1. data collection

2. data cleaning and preparation

3. selection of suitable algorithm/s

4. (optional) feature engineering
5. model training and hyperparameter optimization
6. model testing

Before the data can be used in machine learning, a pre-processing step is necessary to reduce the noise and reformat them for the chosen algorithm. Additionally, the data is split into training and testing or training, testing and validation splits. The algorithm is trained on training data, and testing data is supposed to show a real performance of the predictor on previously unseen data. Usually, the algorithms themselves have parameters, called hyperparameters, and these also need to be optimized. The subset of the training dataset, the validation set, is used to compare scores of models trained with different hyperparameters, acting as an independent set, while the testing set is kept aside and used only to evaluate the final predictor.

In the realm of machine learning, data preparation often necessitates feature engineering [43]. Features are essentially numerical representations extracted from the raw input data that encapsulate informative characteristics relevant to the specific task at hand. Crafting effective features can be a significant challenge in the design of machine learning predictors. While large neural networks exhibit a degree of resilience to suboptimal feature engineering, they often demand considerably larger volumes of training data [4]. In biological and chemical research domains, however, data acquisition is frequently a costly and time-consuming endeavor [28]. This very constraint partly explains the recent resurgence of interest in more general, large protein models, which can achieve good performance even with limited training data [59]. Consequently, feature engineering, when judiciously combined with simpler models, remains a viable and advantageous approach for a multitude of applications, particularly in scenarios where extensive training data is scarce.

4.1 Metrics

In machine learning, various metrics are used to evaluate how well the model performs. In the case of classifiers, a common approach it to score how many datapoints are classified correctly. In regression models, the model performance is evaluated by how far off the real values and the predictions are. The metrics selected are highly dependent on the task the model solves, and often new metrics need to be invented for novel tasks. A noticeable example is text generation, where it is not obvious how to compare generated text to expected while taking into account synonyms and ambiguity of natural language. In this thesis, the main focus is on classification, and there are well-defined scoring techniques for this task.

4.1.1 Accuracy

Accuracy is a metric used to evaluate results of binary classification. It calculates the fraction of correctly classified data points:

$$Accuracy = \frac{TP + TN}{TP + TN + FP + FN} \tag{4.1}$$

where TP = True positive, TN = True negative, FP = False positive, FN = False negative.

Accuracy thus produces scores between 0 and 1. However, in the case of unbalanced datasets (there is a larger fraction of one class compared to the other) the result might be misleading. For a typical example of the prevalence of a disease in a population (usually only a small proportion of the population has the illness), if 2% of the population are sick and the predictor always sees everyone as healthy, it will have the accuracy of 98%, despite not using any useful information in the features.

The `balanced accuracy` score is adjusted for the dataset imbalance, but the problem with looking only on the positive predictions prevails. Thus, there are multiple other metrics that put into proportion other combinations of the TP, TN, FP, and FN. Jointly, they provide a clearer picture of the real situation and are more useful for objective analysis. However, the diversity of metrics brings a higher level of complexity to machine learning, making the model engineering harder. Thus, a single metric that reflects all relevant information is desirable.

4.1.2 Matthews correlation coefficient (MCC)

One of such metrics is Matthews correlation coefficient [41] or Phi coefficient, defined as follows:

$$MCC = \frac{TP * TN - FP * FN}{\sqrt{(TP + FP)(TP + FN)(TN + FP)(TN + FN)}} \tag{4.2}$$

.

It measures the quality of binary classification and returns values between -1 and 1, for negative and positive relationships, respectively. It is not dependent on the ratio of the classes in the dataset. However, the results are harder to interpret for values that are not $-1, 0$, or 1.

4.2 Algorithms

The machine learning domain encompasses a diverse arsenal of algorithms, which are then tailored to address specific tasks. These algorithms include support vector machines, decision trees, random forests, logistic regression, gradient boosting, and neural networks, to name a few. While some algorithms exhibit versatility, capable of both classification and regression, others are designed for specific types of tasks.

Linear models serve as the foundation for many algorithms. As the name suggests, they predict values based on a linear combination of input features. In contrast, more advanced models, such as deep neural networks, transcend these linear limitations, possessing the ability to learn and approximate functions of various complexities. However, this very strength presents a challenge: overfitting. Overfitting is a situation where a significant drop in performance is observed when evaluating the model on the test set compared to the performance observed on the training set. Deep neural networks, with their capacity to fit any function, are susceptible to overfitting the training data, leading to poor performance on unseen data. Subsequent sections will delve deeper into the algorithms most relevant to this thesis.

4.2.1 Logistic regression

Logistic regression is a cornerstone technique in supervised learning specifically designed for classification tasks. It excels at modeling the probability of a particular outcome belonging to one of two predefined classes. Unlike linear regression, which predicts continuous values, logistic regression transforms the linear relationship between the input features and the binary outcome into a probability estimate between 0 and 1. This transformation is achieved through the logistic function Equation 4.3.

$$\sigma(t) = \frac{1}{1 + e^{-t}} \tag{4.3}$$

Logistic regression offers several advantages that contribute to its widespread adoption. A key strength lies in its interpretability. During the training process, the model assigns weights to each input feature. These weights provide valuable insights into the relative influence of each feature on the model's prediction. Features with larger weights exert a more significant impact on the model's output probability, allowing to gain a deeper understanding of the factors driving the classification process. Additionally, logistic regression exhibits robustness to outliers in the data, making it less susceptible to the influence of extreme data points.

However, it is essential to acknowledge the limitations inherent to logistic regression. One such limitation is the assumption of a linear relationship between the input features and the output class. This linearity assumption can restrict the model's ability to effectively capture complex, non-linear relationships within the data. Furthermore, logistic regression can struggle with high-dimensional data containing a vast number of features. In such scenarios, alternative classification algorithms, such as Support Vector Machines or Random Forests, might be more adept at handling the increased complexity.

4.2.2 Neural network

Neural networks, inspired by the structure and function of the human brain, are a powerful tool in the field of machine learning. These computational models consist of interconnected nodes, often referred to as artificial neurons, that process information by transmitting signals between each other. Each connection between neurons has a weight assigned to it, which determines the strength of the signal transmitted. Through a process called learning, these weights are iteratively updated based on minimizing a particular loss function, which depends on the training data and current predictions. This allows neural networks to identify patterns, make predictions, and perform complex tasks without explicit programming.

Critical to this process are activation functions. These functions act as gatekeepers, determining whether a neuron's output signal should be sent forward to the next layer in the network. Different activation functions introduce non-linearity into the network, allowing it to model complex relationships between input data. Common activation functions include the sigmoid function, which outputs a value between 0 and 1, and the rectified linear unit (ReLU), which outputs the input directly if positive and zero otherwise. The choice of activation function depends on the specific task and network architecture.

Neural networks come in various architectures, each suited for specific applications. They have revolutionized various fields, including image recognition, natural language processing [32], machine translation, and bioinformatics [34].

4.3 Large language models

Large language models (LLMs) are at the forefront of artificial intelligence currently, pushing the boundaries of human-computer interaction. These complex algorithms are revolutionizing the way we process and generate text. They are trained on a vast digital library of textual data containing books, articles, social media conversations, etc. The models analyze the training data during training and find patterns and relationships between words. The large sizes of the models allow them to capture grammar, syntax, and other language nuances in great detail.

The true power of LLMs lies in their versatility. They are capable of generating text based on input prompts, essentially allowing users to specify what they need using a natural language, and the users get responses in the same language, in many task types comparable to human responses. The output of the model can also be customized by introducing configuration prompts before the user queries. The LLMs are also capable of condensing and sifting through large amounts of text, as well as extracting key points and creating summaries. They are also capable of seamlessly operating in many languages, allowing users to use them for translation. Another capability is answering factual questions, which comes from the vast amount of information included in the training data. Thanks to these features, the LLMs have quickly become the basis for many personal assistants, internet search enrichment, etc.

In terms of training, there are usually multiple strategies involved for different tasks. One of the tasks is filling in the blanks in the input. The model gets a text or a sentence with some words missing on the input and tries to fill in the blanks. The words the model filled in are compared to the actual words that were masked, and the model weights are adjusted accordingly. This method of training is called **masking**, and the idea is that the model learns language rules and grammar using this objective.

Another strategy is called **next sentence prediction**. In this case, the model is presented with two sentences or two short texts, and the goal is to predict if the second sentence should follow the first one. This objective helps the model to understand context and relations between sentences. This task is more focused on classification tasks, where it is necessary to condense information about the sentence into a vector, so the model has context based on which it can compare the sentences.

Usually, the models do not process text (letters and words) directly, but a preprocessing step is in place, called **tokenizer**. The tokenizer encodes words or parts of words into numerical values, usually integers. It also has to be trained, but the point of training is to find optimal letter groups. The training of the tokenizer is done before training the model, and it is usually fast. Compared to other methods of machine learning, this approach basically avoids engineering any additional features. This burden is on the model during training, which accelerates the process of data preprocessing, but necessitates larger training datasets.

4.3.1 Transformers

The idea behind the transformer comes from a recurrent neural network encoder-decoder [6], where one neural network (encoder) encodes an input sequence into a fixed-length vector representation and another one (decoder) decodes the representation to another sequence of symbols. These two networks are trained together to maximize the probability of the target sequence, given the source sequence.

A recurrent neural network (RNN) is a type of architecture used to process sequential input data. There are many modifications of RNNs, such as Long-short term memory (LSTM) [25] or Gated recurrent units (GRU) [6]. They are still being widely used, with more than 16000 citations on Google Scholar for the original LSTM article alone. In RNNs, the input sequence is processed one token after the other, each token producing a vector representation that is then used when encoding the next token. This brings the inherent sequential processing into training, hindering any useful parallelization possibility and necessitating long training times for large datasets. Additionally, the models tend to „forget“ parts of the previously seen sequence, sometimes parts that are far off from the current token due to vanishing gradient problem [24] and sometimes due to catastrophic forgetting [58].

The term **transformer** was first introduced by [64]. It is also an architecture based on encoder and decoder, but the main difference is that the recurrent mechanism is removed. The input is processed by the encoder as a whole, giving the encoder a view of the entire input sequence. Additionally, it is enriched by a mechanism called attention, which gives each input token a different weight, based on the surrounding tokens, making it possible to highlight important parts of the sequence and ignore the others. More precisely, it is called self-attention, referring to relations between different positions (tokens) of the input. This architecture was shown to excel at text-processing tasks, such as translation [37] and text generation [63].

The encoder consists of multiple layers, each consisting of a multi-head attention layer and feed-forward network layer. There might be a slight variation in how the encoder processes the tokens: some encoders are unidirectional, meaning the current token is processed using attention, taking into account only previous tokens, or bidirectional where during the processing of the current token, all other tokens are taken into account in the attention mechanism. Multi-head attention means that multiple attention-based mechanisms are used in parallel, each focusing on a different importance of the context. The number of layers in the encoder is a hyperparameter that has to be decided before training the model.

The decoder is also formed from multiple layers, and each layer has three parts: attention on the previously generated outputs, attention on encoder outputs, and the feed-forward layer. The first attention layer takes the tokens that the decoder produced previously. When the decoder starts generating a text, it generates the first token. When generating the second token, the decoder already takes the first token into account, etc. This allows the decoder to see what it has generated before, so the following text is concise and well-connected. At the same time, this makes it possible to generate longer or shorter texts than the input, making the length of the output text independent of that of the input.

The second attention layer takes the output of the first attention layer and the output from the encoder as input. The reason is that the decoder still considers what was on the input when generating the answer. The output from the encoder stays the same while the decoder generates the output.

BERT

The Bidirectional Encoder Representations from Transformers (BERT) model was first introduced in [13]. It took only the encoder part of the transformer, and the entire training was done on the encoder only. The goal was to provide a universal model that understood language and could be fine-tuned for various tasks. BERT was originally trained 50% on the masking task and 50% on the next sentence prediction task. Both of these tasks required unlabeled data, which helped with the data preparation and collection.

Fine-tuning is a technique used to adapt large language models (LLMs) to specific tasks. A significant portion of an LLM's capability lies in generating features relevant to the final task. Since this feature generation machinery is reusable across various tasks, it is inefficient to retrain it for each new purpose. Consequently, pre-trained models like BERT exist, focusing solely on feature generation without targeting a specific task.

Fine-tuning leverages these pre-trained models by adding a new layer at the end, specifically tailored to the desired task. This final layer is then trained (fine-tuned) on a dedicated dataset. This dataset can be substantially smaller compared to the massive datasets used for pre-training the LLM. This efficiency arises because fine-tuning focuses on training just the added layer and making minor adjustments to the pre-trained model's internal parameters. The advantages of fine-tuning include facilitating the creation of custom models with greater accessibility, as well as enabling faster model training and evaluation.

Transformers in bioinformatics

The transformer architecture has also successfully been used in bioinformatics for protein sequences [16]. The idea is that the model learns the language of proteins. Usually, the proteins are represented as sequences of letters, where each letter represents an amino acid. In the sense of transformers, each letter is a separate token, and a single protein sequence acts as input to the transformer. The ProtTrans transformer model was, according to this ideology, achieving state-of-the art performance on multiple tasks, such as per-residue secondary structure prediction or missing residue prediction. The authors did not train only transformer models, but also BERT-based models. Analysis of the embeddings with clustering methods implied a high capacity for learning. This conclusion was derived from the analysis of the identified clusters and their mapping to protein properties, such as origin organism, amount of secondary structures, or chemical properties of amino acids.

The AlphaFold [28] model uses attention mechanisms to process multiple sequence alignment and the OmegaFold [68] model uses as a part of its pipeline language model based on transformers. Both of these models are focused on predicting tertiary structures of proteins for their sequences.

Evolutionary scale modeling (ESM2) [65] is another language model based on BERT architecture. It was trained on protein sequences from the UniProt database and has been used for many applications, such as predicting tertiary structure [36] and de-novo protein design. It has also been widely used by the community for various downstream tasks, such as peptide screening [14], protein function prediction [35], or epitope prediction [7].

4.3.2 Fine-tuning LLMs on small peptides

In the recent paper [21], multiple language models were fine-tuned on peptide datasets. The base model was chosen as state-of-the-art transformer ProtBERT [16]. Models were trained on three tasks targeting peptide properties, including solubility, hemolysis, and non-fouling. Peptides are short chains of amino acids, typically containing fewer than 50 amino acid residues. Peptides occupy a middle ground between simple organic molecules and proteins in terms of size and complexity. They are more flexible than smaller organic molecules due to their chain-like structure but lack the intricate folding patterns observed in proteins.

In the paper, the authors used ProtBERT, a transformer model influenced by the original BERT model. ProtBERT is pretrained on the UniRef100 dataset containing millions of unique protein sequences. After the layers used in the base model, a linear layer was added with the sigmoid activation function to perform the binary classification task. The authors trained separate models for each task and showed that the PeptideBERT model had superior performance on two out of the three tasks compared to several baseline models, such as a LSTM, engineered embeddings with logistic regression, or random forests. The model did not perform better only in the solubility task, where DFResSol [39] was the leader. This approach shows promise in adapting a similar strategy to epitopes since epitopes also have a form of short peptides.

4.4 Machine learning in immunoinformatics

As machine learning is becoming more and more prevalent in everyday life, it is also being used in the immunology field for various purposes. The main area of use is vaccine design and development, necessary to combat new mutations in existing viruses as well as new viruses. An excellent example is the COVID-19 vaccines, which were developed in an unprecedentedly short time (around 1 year). For comparison, vaccine development usually took 5 or more years before [51]. Admittedly, the speedup was a result of multiple factors, such as relatively new vaccine technology (mRNA vaccine) and a shortened legislative process, but the recent development in bioinformatics tools and machine learning technologies also played a significant role [26] [48] [52]. Such tools include complex tools for designing vaccines, such as Vaxign2 [46], but also simpler tools designed to analyze epitopes [53] [50] [7] [60].

4.4.1 Epitope prediction

In epitope prediction, the goal is to predict parts of antigens that are causing the immune reaction – epitopes. Thus, an ideal tool would be able to accurately identify potential epitopes in a protein sequence with high sensitivity and specificity. However, as explained earlier, the interactions of the immune system with antigens are highly individual. Thus, the tool should be able to generalize to how most of the population would react to an antigen.

Machine learning in this field proves to be a major challenge, as the immune system is very complex and difficult to measure. Nevertheless, a substantial number of methods have been published, and some are available as web-accessible tools. Usually, each subsystem of the immune system is addressed separately, as a reliable method for complex prediction of the immune response is not yet available. Each of the subsystems has different stages of tool reliability and quality.

For T-cell immunogenicity, a recent artificial neural network-based tool that predicts T-cell epitopes based on MHC binding is claimed to achieve a 90% correlation [54] on the test set. The other categories are MHC II binding predictors [29] and B-cell epitope predictors [27] [8]. These are proving to be more challenging, especially because B cells have around 90% structural epitopes, which require exploring the 3D structure of given proteins and the 3D structure of their binding to antibodies. B cells also produce antibodies, which might inhibit the target therapeutics. Thus, the B cell epitope engineering is the main motivation for collecting mutational data.

The Immunoepitope Database (IEDB) [66] serves as the primary source for epitope information and is currently the most extensive database available. It houses data from thousands of assays encompassing B-cell, T-cell, and MHC-binding immune responses. Epitopes within IEDB can be derived from proteins, pathogenic molecules, such as organic compounds, or even glycoproteins. The database meticulously stores results for each potential epitope, including its corresponding measured immune response (positive or negative). The data can be downloaded in a user-friendly SQL format, with well-documented table relationships visualized through Entity-Relationship diagrams. For structural information on antibody-antigen interactions, AbDb (or abYbank) [18] offers a complementary resource.

Currently available tools primarily focus on predicting the structural impact of mutations, with limited capacity to address the effect on protein immunogenicity or epitope presentation. This knowledge gap stems from two main factors. Firstly, there is a scarcity of well-curated datasets specifically designed to address the relationship between mutations and immunogenicity. Secondly, a significant portion of research in this field prioritizes the development of effective antibodies against pathogens, rather than proteins that evade immune response. Developing such tools is crucial for the design of novel protein-based therapeutics with minimal immunogenic response. Additionally, these tools could be instrumental in creating improved vaccine delivery systems and targeted drug carriers that evade clearance by the immune system.

4.4.2 Vaxign2

Vaxign2 [47] is a web-based vaccine design program that leverages machine learning and reverse vaccinology methods to identify potential vaccine candidates. It analyzes pathogen genomes, including those from viruses and bacteria, to propose suitable targets for vaccine development. Vaxign2 has been instrumental in suggesting novel vaccines for various diseases, successfully pinpointing the spike protein as the primary candidate for a COVID-19 vaccine. The program's machine learning component relies on models based on gradient boosting. It is important to note that Vaxign2 is specifically focused on target identification for vaccine development and is not intended for epitope analysis or protein mutational design.

4.4.3 BepiPred

BepiPred represents a successful suite of tools designed for B-cell epitope prediction. The latest iteration, BepiPred-3 [7], leverages protein structure data from the PDB database alongside the power of language models to predict structural epitopes. BepiPred-3 incorporates the ESM2 model [65] developed by Facebook AI, fine-tuned to generate scores between 0 and 1 for each residue in a protein sequence. These scores indicate the likelihood of a residue being part of an epitope (values closer to 1) or not (values closer to 0). The tool also provides user-friendly visualizations to aid in interpreting the predictions.

Earlier versions of BepiPred also focused on structural epitope prediction, with version 2 employing a random forest model. The original version, published in 2006, addressed linear epitope prediction using Markov models.

Predicting the effect of mutations is highly beneficial for protein engineering. A multitude of predictors exists for predicting effects on stability [44] [5] [55], or activity [19]. We found out that the existing tools for predicting immunogenicity, such as BepiPred3, do not perform well for mutational effects (Table 7.1 and Figure 7.3), thus a novel tool for such task would be advantageous in designing novel therapeutics.

Chapter 5

Data collection

In order to create machine learning-based tools, it is necessary to create or find a suitable dataset for a given task. For effective prediction of mutation effects, usually, the dataset consists of different single or few-point mutations in a single wild-type protein sequence, or if the collection is larger, on a set of wild-type sequences. For non-mutational datasets, each datapoint consists of a protein sequence and/or its structure and such labels as melting temperature, activity for enzymes, etc.

The field of immunoinformatics encompasses a diverse range of tasks, including predicting antibody-antigen binding, designing antibodies for specific antigens, paratope similarity, and epitope prediction. This thesis focuses on immunogenicity prediction, which aims to determine whether a given epitope or peptide will elicit an immune response. Although certain types of experiments are capable of measuring the extent of the immune response, most of the datasets are based on binary labels, labeling epitopes as inducing or non-inducing immune response.

5.1 Existing data sources

To our knowledge and according to our research, no tools or large, multi-protein data sets have been published focused on the effect of antigen mutations on immunogenicity. The existing datasets are small in size (up to 100 datapoints) and specific to a particular protein. Since acquiring this kind of data experimentally is not feasible, it is necessary to obtain the data in a different way.

It is important to note that around 80% of B-cell epitopes are structural in vivo. An ideal predictor would focus on the structural epitopes. However, in BepiPred3, the authors collected almost all available structural data of antibodies bound to antigens, resulting only in 1466 datapoints. Considering the possible space of structural epitopes in comparison to linear ones, it is unlikely that enough mutational datapoints could be mined from this dataset following a similar protocol as suggested in this chapter. Additionally, working with structures brings another level of complexity to data mining and the machine learning approach. Since the field of mutational effects on epitopes has not been explored much, it is reasonable to start with linear epitopes. Thus, in the rest of the thesis, our discussion will center on linear epitopes only.

5.1.1 Staphylokinase study

One of the studies that attempted engineering of an existing protein as a therapeutic, studied staphylokinase protein from a bacteria *Staphylococcus aureus* [11]. It is a thrombolytic enzyme, which means it can degrade blood clots or aid in their degradation. During development, the authors also performed immunogenicity studies. Since they engineered more than 100 different staphylokinase mutants and tested almost all of them for immunogenicity, the study presents a valuable source of data. The dataset size is still not enough for training a neural network, and it covers only one protein. However, it can be used as an independent data set to test existing and new predictors. This data had to be obtained from multiple publications and patents. Apart from immunogenicity labels, the dataset also contains enzyme activity labels, which are also valuable for other machine-learning tasks. This study is the motivation behind the creation of a mutation-based predictor of immunogenicity.

5.1.2 Immunoepitope database

The largest database of immunogenicity effects on antigens is **IEDB** [66]. It contains epitopes from thousands of assays encompassing B-cell, T-cell, and MHC-binding immune responses. Usually, immunogenicity assays for protein linear epitopes are based on splicing the protein into small peptides and screening those peptides for the immune response. The database is conveniently separated into different parts of the immune system, such as B-cell receptors, T-cell receptors, and MHC bindings. For the development of novel protein-based drugs, it might be necessary to optimize for B-cell and T-cell epitopes. However, for acutely administered drugs, the B-cell immune response is more important, more specifically the antibody binding, since the antibodies can disable the drug just by binding to it. Thus, from now on, we will focus on B-cell epitopes, although a similar protocol can be utilized for other types of immunogenicity as well.

5.1.3 Mutational data

As mentioned above, IEDB contains epitopes and their immunogenicity, but no mutational data. So, the main question is how to obtain the mutational data from the non-mutational database. Often, IEDB entries originate from the research of viruses and other pathogen-like antigens. These organisms tend to mutate, especially viruses, and some of these mutations might end up being viable for avoiding the immune system. Such mutations usually tend to lead to more intensive spreading of the virus and spark up the research again. In this case, the mutated antigens are investigated and screened again, but the protein sequence that is saved in the database with the assay results is already slightly changed because of the mutations. By utilizing this natural cycle, we can take advantage of the results and obtain these mutations from the database. Additionally, we can also focus on specific epitopes if the epitope was deactivated or a new epitope emerged from mutations.

Thus, the idea for obtaining a mutational dataset is to cluster these mutated proteins based on their sequence similarity and to create groups of the same proteins with different mutations. Then, the epitopes of the proteins within the group are scanned, and the mutations are examined that cause the loss of immunogenicity for a given epitope or the emergence of a new epitope. By taking advantage of available data and processing them in a different way, it might be possible to create a new mutational dataset without any new experimental work necessary. This new dataset can be used later for training or fine-tuning of models or statistical analysis. To my knowledge, no such dataset exists yet.

5.2 Technical steps

The following steps outline the retrieval of IEDB data and the essential data-cleaning procedures employed. The protocol is written for B-cell epitopes, but a very similar approach can be taken for T-cell epitopes.

5.2.1 Database processing

The IEDB has to be downloaded in SQL format and imported to a local database engine. Then the structure of the database has been analyzed with the help of the ERD diagram provided by the IEDB to understand which tables are necessary in the next step. The database has no foreign key constraints, only pure ID references to other tables, and in certain tables even the IDs are not unique. This poses a problem, and uniquification of tables is required in the next steps.

Table `bcell` contains the results of the B-cell receptor assays, and each item has its ID and reference to the curated receptor. The results are also reported for different host organisms, and in our case, we are only interested in the human host. Since multiple assays can explore the same epitope, the curated receptor IDs are used to reference the same epitope in different assays. Sometimes, assays can have contradictory results. Thus, the first step is to count the number of positive and negative assays for each epitope and to exclude epitopes with contradictory immunogenicity results. This step was done using SQL queries.

The next step is to filter only protein epitopes since the database also contains other types of antigens. However, the `bcell` table does not contain data for the antigen source molecule directly, only as a reference to the `source` table. Unfortunately, this table does not have a unique index, and some entries are duplicated multiple times. The table has to be deduplicated, and subsequently, the tables can be joined. Finally, the resulting data can be exported into a CSV file to the extent of:

- Result of the assay
- Epitope information – curated epitope ID, epitope sequence, position in the protein
- Protein information – protein sequence

5.2.2 Data refinement

The previous step yielded only filtered epitopes and not a mutational dataset, so further processing is essential. In the following steps, the data will be processed using Python (3.9) and Pandas library (2.2.1) and saved in a binary format using Pickle library, instead of using CSV format due to superior loading speed and keeping track of data types.

After a short manual exploration of the database export, multiple inconsistencies in the data have been detected:

- There are duplicates in the data,
- The entire source protein sequences are missing for some entries,

- The coordinates specifying epitope locations within the sequence are absent or erroneous.

The first two problems can be easily addressed by deduplication and removal of the epitopes with missing sequences. The third one is more elaborate and requires more effort, since it would be unreasonable to remove entries affected by this problem. Furthermore, the epitopes are still not grouped in a way that similar proteins are in one group, which is necessary to produce mutational data.

First, in order to group similar sequences, the `mmseqs2`[62] tool is used to create clusters of similar protein sequences. The sequences need to be exported from the data, and ideally, only unique sequences are clustered, since in the data exported from the IEDB, multiple epitopes are present for each sequence. Fortunately, each sequence already has a unique ID, so a FASTA file is created with the ID of each sequence in the description. After tuning the parameters of `mmseqs2` and by manually exploring the results in `Jalview` program, the most promising values are established at the 80% sequence identity with the 80% coverage. The following shell commands were executed to cluster the sequences, which are stored in `bcell_db.fasta` file:

```
$ mkdir tmp
$ mmseqs createdb bcell_db.fasta
$ mmseqs cluster -c 0.8 --min-seq-id 0.8 bcell_db bcell_clustered tmp
$ mmseqs createtsv bcell_db bcell_db bcell_clustered bcell_db_clustered.tsv
```

Next, the sequences within each cluster need to be aligned to have a global coordinate space within the cluster. This is necessary to compare different epitope positions between sequences to properly create the mutational dataset. For each cluster separately, a `MAFFT`[30] alignment tool is run. Afterward, the aligned sequences are imported back into Python and merged with the corresponding epitopes together with the newly obtained group IDs from `mmseqs2`.

At this point, there is a problem with the epitope coordinates in the sequence because most of the sequences now have inserted gaps. The coordinates for each epitope have to be recalculated to match the aligned sequence. Concurrently, if the coordinates are missing because they were not provided in the IEDB, Algorithm 1 attempts to fill them with multiple strategies. Some epitopes have the coordinates as part of the epitope name, so the algorithm tries its best to extract them and only then it tries searching the epitope in the protein sequence. If the newly-found position is ambiguous because there might be multiple positions where the epitope occurs in the protein sequence or the epitope was not found at all, the datapoint is discarded.

In order to accelerate the process, these steps are done in parallel for each cluster, and then the results are merged into one data frame. The MAFFT tool was used for its superior speed compared to CLUSTALW and also because it is expected that the clustered sequences are similar, and thus much easier to align.

Clusters with only one unique sequence are removed, which automatically discards clusters with one data point. Moreover, we will need to create contrastive data points, that is, a set of mutations that lead to the decrease/increase/not change in the epitope. This can only be done within clusters. Therefore, only clusters with both positive and negative data points are left in the dataset and the others are filtered out.

Algorithm 1 Region coordinates validation and reparation

```
 1: if epitope has coordinates then
 2:     verify coordinates by matching the sequence to the epitope
 3:     if the sequence does not match then
 4:         try looking at 5 amino acids around the given position, in case it was mislabeled
 5:     end if
 6: end if
 7: if the previous step did not produce valid coordinates then
 8:     if the coordinates can be obtained from the name of the epitope then
 9:         obtain the position
10:     else
11:         search the protein sequence for the unique position of the epitope
12:     end if
13: end if
14: if some valid position was found then
15:     recalculate the position relative to MSA
16: end if
```

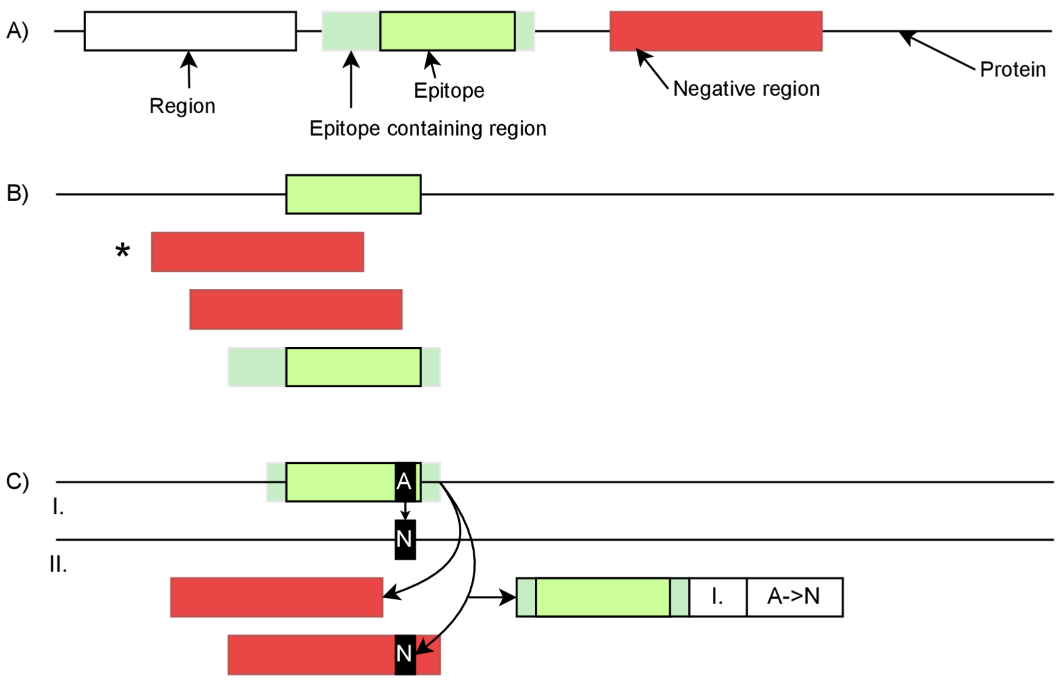


Figure 5.1: **Region explanation.** A) Visualization of terminology. Some regions correspond directly to epitopes, but some are larger parts and span around the epitopes – epitope-containing regions. B) Example of how negative regions can contain a part of the epitope but have a negative immunogenic label. *The regions are all on the same protein but are shown below for better clarity. C) The process of generating the final data points. Two regions are considered, each from a different source sequence. One has to be positive and one negative. If the negative epitope is longer or the same size as the positive epitope, **completely** contains the region of the positive epitope, and differs in only one or two amino acids, then a new data point is created, marking the mutation as reducing immunogenicity.

5.2.3 Final dataset creation

It is important to stress at this point that not all datapoints obtained from IEDB are precise epitopes. Some of them are epitope-containing regions (ECRs) and some are exact epitopes (EEs) (see Figure 5.1). Each epitope has a label that indicates its type (ECR or EE). The epitope-containing region is a short part of a protein (up to 30 amino acids) and contains an epitope inside, but it is not precisely specified where the epitope is within this region. The exact epitope region means that the entire region is an epitope. However, for the initial creation of the dataset, this distinction is not taken into account but could be considered if the dataset proves to be too noisy.

In general, it is not strictly correct to call all the entries obtained from IEDB epitopes, since the epitope, by definition, is the region that induces an immune response, so only positive data points that are exact epitopes are truly epitopes. The negative entries should be called non-epitopes. Therefore, from now on, we call all the entries **regions**, no matter if they are positive, negative, ECRs, or EEs.

To create the contrastive data points, the data set is iterated by clusters and, within each cluster, the regions are assigned to negative and positive classes based on the immunogenicity label. For most of the clusters, there are many more regions in the negative class than in the positive class. For each region in the positive class, the algorithm searches through regions in the negative class that contain the positive region coordinates. Then the overlapping parts of the positive and negative regions are compared and the number of mismatches is counted. If the number of mismatches is one or two, then a new mutational datapoint is created with a label marking it as decreasing immunogenicity.

The regions might overlap in different ways, but only these instances or their combination are considered:

- **Equal** – the start and end coordinates are the same for both
- **Containing** – the start and end of the positive region are inside the negative region.

A similar process is performed for only positive regions, where one positive region is compared with other positives within the cluster. In this case, the new mutational datapoint has a label marking it as non-changing, meaning the mutation did not change the immunogenicity. After these two steps, it is necessary to drop duplicates because it is possible that the same region with the same mutation was marked as immunogenic and non-immunogenic.

5.2.4 Encountered problems

When epitopes are evaluated experimentally, the proteins are sliced into smaller pieces, and the results contain almost all possible slices of a given length, the lengths of the regions being constant or very similar within one assay, even if just a part of a protein is screened. Thus, there is often a large overlap between the screened regions.

At first, we considered any overlap of positive and negative regions, and the union of the two regions was investigated as well. However, with such a high overlap rate between regions, artifacts were created most of the time, where the same mutation was labeled as inducing immunogenicity and suppressing it at the same time. This was because the regions were labeled as causing immune reactions only when the epitope was completely contained in the region. The regions that did not contain the entire epitope and thus did not induce an immune response were labeled negative. Therefore, a slight shift of the slicing window created non-immunogenic regions and many false immunogenicity-reducing mutations (see Figure 5.1). Thus, only the negative regions that entirely covered the positive region were considered, and this is captured in the overlap types mentioned above.

The next level of complication is introduced by epitope-containing regions. If a mutation is found in ECR, there is a chance that the mutation occurs outside the epitope, but still within the region, so it does not affect immunogenicity in reality but would appear so in the dataset. The corresponding mutation would become part of the dataset and would be labeled as negative. It would be on the same level as mutations that truly decrease immunogenicity. The only way to mitigate this would be to exclude ECRs from the initial set. However, this would lead to a major loss in dataset size, so for now, this problem is ignored and can be addressed later.

Algorithm 2 Mutational dataset procedure

```
1:  for cluster in input dataset do
2:      for pos in positive samples of cluster do
3:          c1 ← does not come from the same sequence
4:          c2 ← contains the whole pos region
5:          negs ← negative samples from cluster that comply with c1 and c2
6:          for neg in negs do
7:              if pos and neg have 1 or 2 different amino acids then
8:                  add this datapoint to the final dataset
9:              end if
10:         end for
11:     end for
12: end for
```

5.2.5 Data collection results

After the first step of getting the data from the database, 432 770 rows were obtained. These were epitopes that did not have conflicting assay results. These epitopes come from 9411 source sequences. The source sequences of these epitopes were clustered together, resulting in 4009 clusters. After filtering out epitopes that were not in clusters with positive and negative samples present, the number of clusters was reduced to 495, and the number of unique source sequences was reduced to 4676. The distribution of their sizes can be seen in Figure 5.2 and Figure 5.3. Overall, there were 202 837 entries left after filtering and clustering.

The mutational immunogenicity dataset includes 8742 datapoints with 1584 of them labeled as reducing immunogenicity and 7158 labeled as not changing immunogenicity. This makes the immunogenicity reducing datapoints only 18% of the entire dataset. In the entire data set, there are 4407 unique sets of mutations per cluster, and this number gives some intuition on how many epitopes occur in multiple ECRs or the same epitope in different protein sequences, so the same position, and thus mutation, occurs multiple times among the data points. However, this does not necessarily mean data duplication, since the sequence of the region itself is different, so the data points with this property are kept in the final dataset. In terms of mutation count, the data set is made up of 5694 single and 3048 double-point mutations. The number of datapoints contributed by different-sized clusters can be seen in Figure 5.4. The distribution of individual epitope sizes can be seen in Figure 5.5.

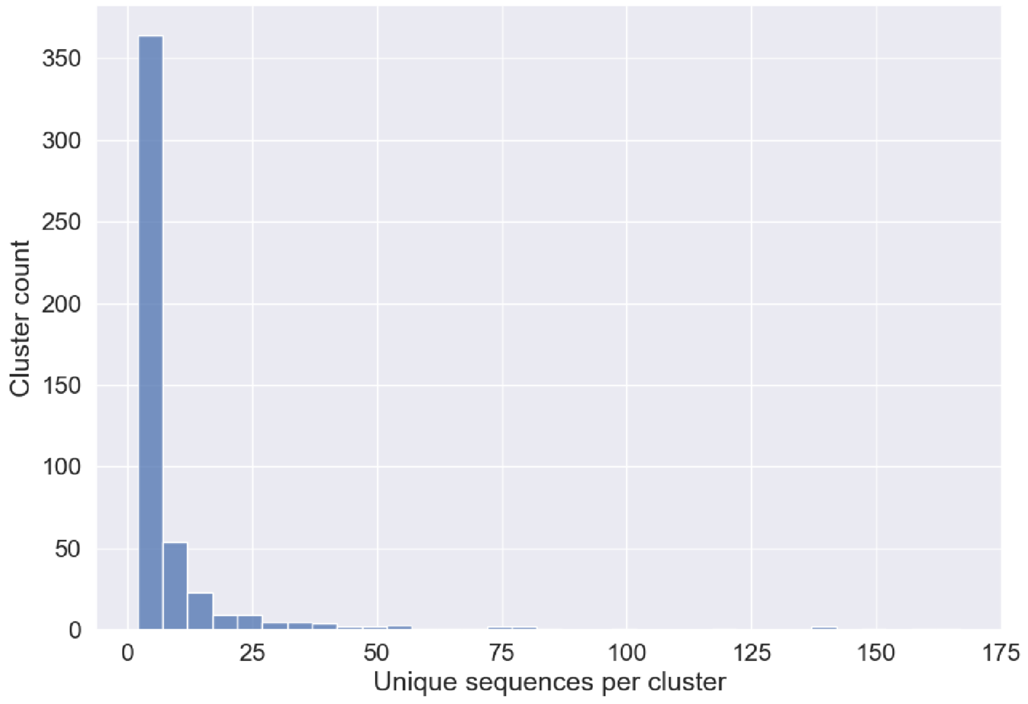


Figure 5.2: Number of unique sequences in clusters. It can be seen that most of the clusters have fewer than 12 unique sequences.

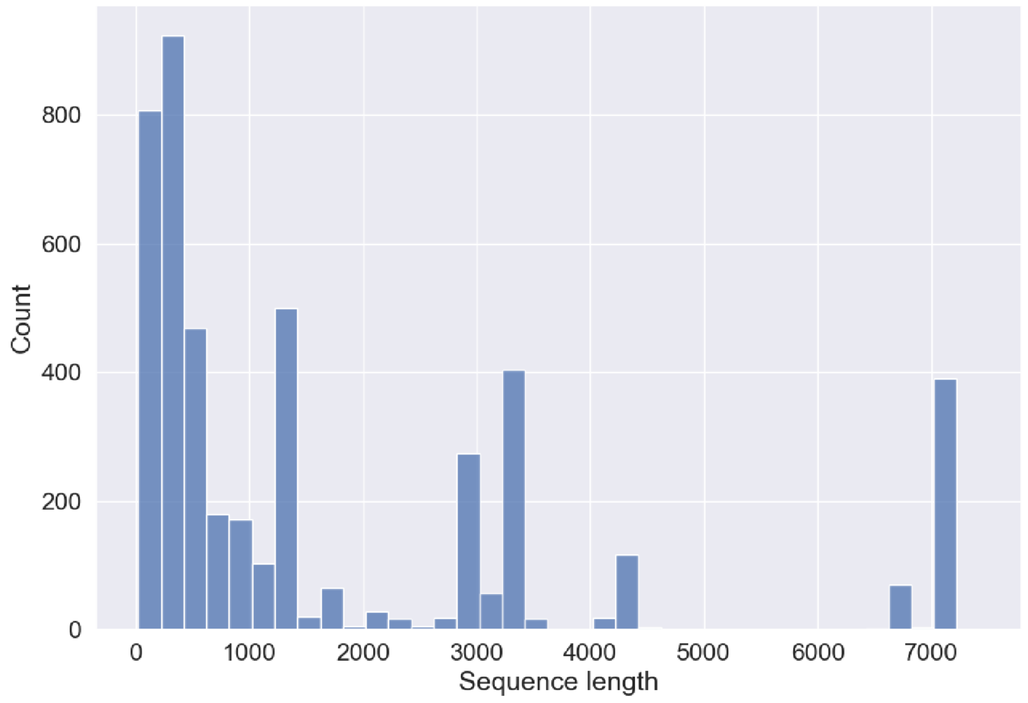


Figure 5.3: Distribution of the lengths of unique sequences. Around 30% of sequences are longer than 2000 amino acids.

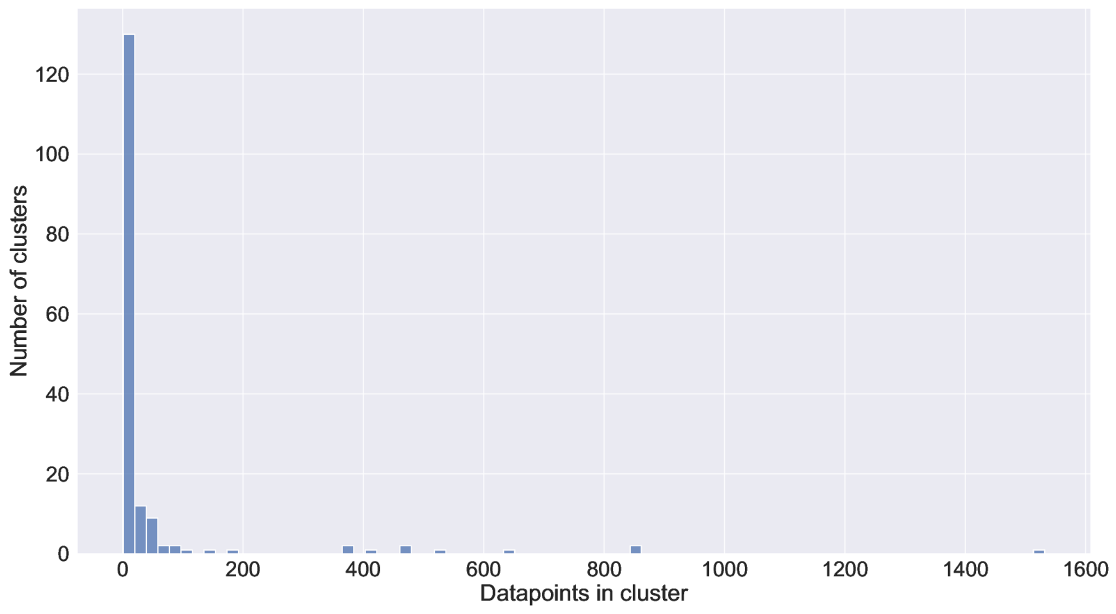


Figure 5.4: Number of data points contributed by the individual clusters. Most clusters contribute very few data points. The largest cluster contributes around 17.5% of all data points, and this is a cluster of sequences of hepatitis virus C.

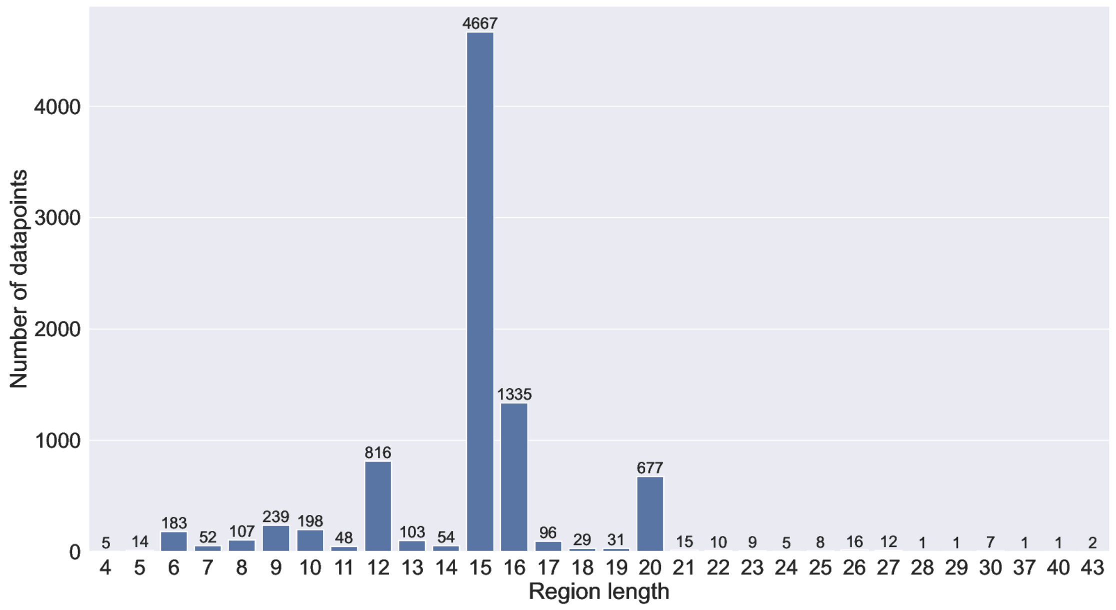


Figure 5.5: Region size distribution in the final dataset.

Chapter 6

Models

As mentioned in section 4.4, there are no tools focused on predicting the effect of mutations on the immunogenicity of epitopes. However, existing tools for predicting epitopes from protein sequences can be adopted to predict the effect of mutations by evaluating the original sequence and the mutated sequence and comparing the scores. In addition, we introduce a new predictor that is focused on evaluating whether a given mutation reduces immunogenicity or not. The first step in a machine learning pipeline, after collecting the data, is to create a data split.

6.1 Splitting the dataset

To properly evaluate existing tools on this dataset and to create a new tool, it is important to split the dataset into training, validation, and test splits. The training and validation splits will be used to train a new predictor and optimize its hyperparameters. The testing set will be used to benchmark the developed predictor against simple baseline methods and the existing tools. The distribution of datapoints into training, validation, and test was decided to be $75\% - 15\% - 10\%$ respectively.

6.1.1 Random split

The simplest way to divide the data set is to randomly distribute the datapoints. However, this would create a split with a high level of leakage, and scores obtained from validation and test splits would not be representative of real-world performance. Indeed, there is a high probability that the same epitope that is contained in different regions would appear in different splits. This is not desirable and would artificially boost the measured performance of the new tool. Thus, a more elaborate strategy is needed. However, for comparison purposes, we created this split to see how the performance of the predictor would differ compared to more elaborate splits.

6.1.2 Split based on protein sequence similarity

The first variant of the split is based on the sequence similarity of the source protein. The idea is to keep the epitopes that come from the same protein in the same split. Clusters that were computed during initial steps of dataset creation using MMseqs (see subsection 5.2.2 are a good starting point for this objective. However, certain sequences form separate clusters, even though they have a high similarity since the clustering was done for a relatively high similarity threshold of 80%.

For this reason, a new clustering is performed on sequences that represent each cluster, but in this case, the objective is to obtain the ancestral tree with distances between nodes and leaves. Hierarchical clustering is done by the ClustalW[1] tool and exported in Newick format. The BioPython library (v1.78) is used to parse and traverse the Newick tree format. After a manual inspection of the tree, a clustering threshold is chosen empirically as 0.5, based on the distances of similar sequences in the phylogenetic tree.

An algorithm traverses the tree and creates an undirected discontinuous graph where the sequences from the tree represent vertices, and if they have a distance smaller than the threshold, an edge is formed between them. This graph is then traversed, and all continuous subgraphs are found, and they represent a *supercluster* of sequences that should be kept together within one split. These superclusters form basic units that will be distributed into the splits. However, now the units that have to be distributed between the splits differ widely in size. Additionally, there is the need to keep the splits a certain size.

This leads to an optimization problem called the **knapsack problem**, where items of different values need to be put in a place with limited available space. Each item has two attributes: weight and value. The knapsack has limited carrying capacity. The goal is to find a combination of items that provide the highest value and still fit into the backpack. In this case, the *superclusters* represent the items, and their value is the same as weight and equals the number of datapoints that these superclusters contain. The knapsack represents the split (test, validation), and its carrying capacity is the size of the whole dataset multiplied by the percentage that should be in the split. For example, the carrying capacity for the validation split would be $8742 * 0.15 = 1311$ (total dataset size $*$ validation set size percentage).

The algorithm is run twice, for validation and the test sets. The remaining *superclusters* are assigned to the training set. The implementation of the knapsack solver is used from the library ortools[2] (v9.8) provided by Google.

6.1.3 Split based on epitope similarity

Although the previous method provides a solid separation, for predictors that do not consider the source sequence, it might not be the best strategy because similar epitopes might end up in different splits if different proteins contain similar parts or domains. A better separation would be based on epitope similarity, which allows a more direct and cleaner grouping of similar epitopes. The protocol is similar to the previous technique, with slight modifications.

First, the similarity map is computed for all regions in the dataset, which means comparing each region to the others by aligning them using the global alignment method implemented as PaiwiseAligner from the BioPython[3] library with `BLOSUM62` score matrix. If the compared epitopes are **aligned with up to two mismatches, with up to one gap that is at most one in length**, then they are marked as *close*. An undirected discontinuous graph is then constructed in which the vertices are represented by epitopes and the edges connect epitopes that are marked as *close*. From this point on, the protocol follows the steps of the split based on protein sequence similarity: the graph is separated into contin-

[1] https://www.ebi.ac.uk/jdispatcher/msa/clustalo
[2] https://developers.google.com/optimization
[3] https://biopython.org/

uous subgraphs, *superclusters* are computed, and the knapsack solver is used to construct the splits.

6.2 Baseline models

To establish the baseline performance achievable with simple machine learning approaches, we resort to feature engineering and linear models. To this end, two linear model algorithms were used: a logistic regression and support vector classifier with class balancing, since the dataset is very unbalanced. We also explored three groups of features: one-hot encoding, AAindices, and ESM2 embeddings.

6.2.1 One-hot encoding

In this approach, features are one-hot encoded amino acids of the mutation. For example, if the mutation is I3L, that means that the isoleucine (I) at the third position in a region is mutated to a leucine (L). Each amino acid is one-hot encoded by a vector of 22 (20 typical amino acids, a gap, and an X which marks an unknown amino acid), and the vectors for both wild-type and target amino acids are concatenated to create a feature vector of 44 dimensions. In each 22-long vector, there is one in the position of the particular amino acid and other positions are zero.

6.2.2 AAindex encoding

One-hot encoding does not incorporate physical-chemical similarities between different amino acids. To fill this gap, we also transformed each region into a set of vectors based on the AAindex database, which contains the chemical and physical properties of individual amino acids. In total, 7 AA indices were selected, and wild-type and target amino acids were encoded using these properties. We used two approaches to combine those two vectors: a simple concatenation and taking a difference of these properties between the wild-type and target.

6.2.3 ESM2 embeddings

A more elaborate approach is to use pre-trained language models on the epitope regions, acquire embeddings from the last hidden layer, and use these embeddings as features. This approach takes advantage of the model's understanding of the protein language but does not require any training of neural networks, sacrificing in part the explainability, since a neural network provides the features. ESM2 was used to obtain embeddings of epitope regions in the inference mode, with average pooling over the length dimension of each region.

6.3 Fine-tuning ESM2 for mutational effects

Our more advanced approach is a fine-tuned large language model. ESM2 was selected as the pre-traned model due to its ease of use and wide adoption across various protein engineering tasks.

The first problem that needed to be solved in our case was how to get rid of the embedding size difference arising from the lengths of different regions because the embeddings initially were a two-dimensional matrix for each region, one dimension matching the sequence length and the second dimension being constant. The aggregation step is necessary if we want to use feedforward layers that have constant sizes. The following strategies were used to remove the variable element from the embeddings:

- mean
- max
- mean combined with a standard deviation
- pooling layer with trainable parameters
- embeddings of the [CLS] token of ESM2
- embeddings of the [SEP] token of ESM2

The aggregation functions aggregate across the dimension corresponding to the sequence length. The [CLS] token is a special token added at the beginning of each input sequence. It originates from the original BERT model, where it was used for the next sentence classification task. The [SEP] token is added at the end of each input, indicating the end of the sequence. There is no apparent reason why these tokens should contain information about the input sequence. However, it has been proven that using embedding from these tokens provides good performance in other tasks [9] [21]. One of the reasons might be that the [CLS] and [SEP] tokens are present in every input and always at specified positions, whereas other tokens change, so the model uses the constant tokens to save contextual information about the sequence that is later used during the reconstruction of masked tokens.

Another problem arises from the comparison of the embeddings of wild-type and mutated regions. Three strategies were tested: **Concatenation of the aggregated embeddings.** The embeddings obtained from the previous step were concatenated into one vector and fed to the fully connected network. **Element-wise difference** of the wild-type embedding from the mutated embedding. The result was a vector of the same size as the embeddings that were inputted into the feed-forward network. **Concatenation of input sequences**. Both wild-type and mutated region tokens were concatenated and separated by a special token, similar to the next sentence prediction tokenization in BERT.

Another optimization parameter was the shape and size of the fully connected network. Three variants were tested:

- One layer with a sigmoid activation
- Two layers, with ReLU after the first layer, and a sigmoid after the second
- Three layers, with ReLU after the two inner layers, and a sigmoid after the last layer

The last layer always has only one neuron with a sigmoid activation for classification of immunogenicity.

In the first iteration of experiments, the weights of the ESM2 were frozen and only the feed-forward layers were trained. In the second iteration, all the weights, including ESM2, were trained.

6.4 Third-party tools

As mentioned in chapter 4, there are no state-of-the-art tools for predicting mutational effects on epitopes. Nonetheless, multiple epitope prediction tools exist, and although they are not focused on mutagenic effects, it is possible to adapt them. These tools are usually trained on entire protein sequences and are intended to be used as such.

BepiPred3 [7] provides a per residue scoring of the input sequence in the range between 0 and 1, symbolizing the confidence that this residue belongs to an epitope. It gives two scores for each residue, one for conformational epitopes and one for linear epitopes. As we are not using structures in our workflow, we use only linear epitope scores.

We will test two different approaches in adapting BepiPred3 for mutational effects. In the first approach, the model is given the wild-type and mutated regions separately and evaluates them. Then the scores for each residue are added together for each region, and the wild-type score is compared to the mutated one. If the score is lower for the mutated region, the mutation is marked as reducing immunogenicity, otherwise as non-reducing. In the second approach, the entire protein sequences of the wild-type and mutated proteins are presented, also separately, on the input. The model provides scores for the full sequences, and when evaluating immunogenicity, only the scores for the region in question are considered and evaluated in the same way as before, i.e., by adding them together and comparing the sums between wild-type and mutated regions.

LBTope [61] is an older (2013) tool for evaluating the immunogenicity of peptides. It provides a probability of each input sequence being an epitope or not. This tool was evaluated similarly to BepiPred3, where it was presented with wild-type and mutated peptides on the input, and if the probability was lower for the mutated region, then this region was labeled as reducing immunogenicity.

We have also tried to use the epitope1D tool [60]. However, it was not possible as the tool required taxonomy of the origin organism, and many datapoints from the dataset were not tagged with this information.

Chapter 7

Results

This chapter delves into the results of applying various machine learning algorithms mentioned in the previous chapter to the newly constructed dataset for predicting mutational effects on protein immunogenicity. We analyze the performance of baseline models and third-party tools and compare them to the ESM2-based models, evaluating their ability to accurately classify mutations that alter immunogenicity. The comparison of the created models can be found in the Table 7.1.

7.1 Baseline models

The scores for baseline models can be seen in Table 7.1. The two algorithms, Logistic regression (LR) and Support vector classifier (SVC), are marked accordingly together with the features used. The best performing from the baselines is SVC with AAindex difference between wild-type and mutated amino acids with the MCC score 0.18 and accuracy of 0.61. Dependencies between different features and feature distribution can be seen in Figure 7.1. There is no clear separation of the two classes, and the distribution of the feature values is almost identical between positive and negative classes. This implies that the task of predicting the change in immunogenicity cannot be easily reduced to a simple evaluation of the change in physicochemical characteristics of the mutated amino acid, and more context needs to be taken into consideration when making a prediction. This is also reflected in the scores of the models trained using these features being close to random guessing.

7.2 ESM2-based models

The second round of models consists of ESM2 embeddings of the wild-type and mutated regions, both of them separately aggregated by averaging and using the aggregated embeddings as input features to LR and SVC. These models performed much better, compared to baseline models, achieving an accuracy higher than 70%. The SVC models performed marginally better than the LR models. Using the variant of the ESM2 model with a larger size did not seem to provide additional benefits.

The third round consists of a purely neural-network-based approach. We performed numerous experiments to optimize the parameters and structure of the model. We use the 150 million parameter ESM2 model variant and binary cross entropy as a loss function. The validation set was used to score the different modifications of the models for comparison and parameter optimization. The ESM2 model was extended by linear layers and different

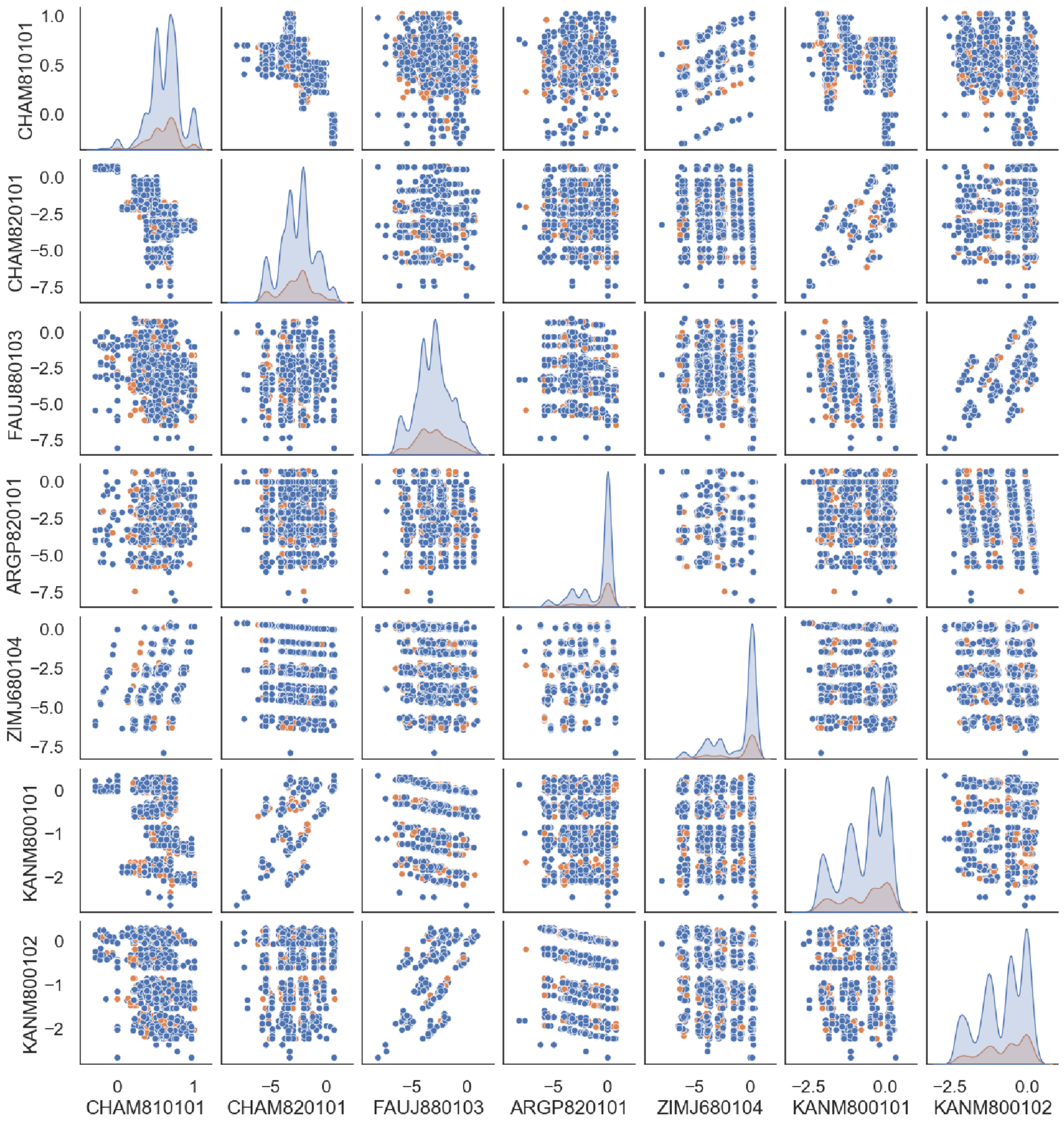


Figure 7.1: Pairplot of the AAindex features where the wild-type amino acid values were subtracted from the target amino acid. The codes of the features are provided next to the axes. It can be seen that there is no clear linear separation of the positive (orange) and negative (blue) labels within the dataset.

Table 7.1: The summary of the performance of all the models on the held-out test set. In general, the logistic regression models performed worse compared to the Support Vector Classifiers trained with the same features. The best-performing model in terms of accuracy was the Support vector classifier with ESM2 embeddings of the entire epitope region, but in terms of MCC and precision it was ESM2 with frozen weights and a linear layer. The fine-tuning of ESM2 yielded slightly worse results than using frozen weights. All the third-party tools showed performance close to random guessing. (The scores are rounded to two decimal digits, but the highlights are based on the highest score without rounding.)

Model	Balanced accuracy	MCC	Precision	F1
Baseline models				
LR with one-hot encoding	0.55	0.07	0.56	0.50
SVC with one-hot encoding	0.58	0.14	0.65	0.46
LR with concatenated AAindices	0.51	0.01	0.51	0.46
SVC with concatenated AAindices	0.58	0.13	0.64	0.47
LR with subtracted AAindices	0.48	-0.02	0.48	0.45
SVC with subtracted AAindices	0.61	0.18	0.69	0.52
Linear models with ESM2 embeddings				
ESM2 150M embeddings + LR	0.72	0.35	0.77	**0.69**
ESM2 150M embeddings + SVC	0.72	0.36	0.78	0.69
ESM2 650M embeddings + LR	0.71	0.33	0.76	0.68
ESM2 650M embeddings + SVC	**0.73**	0.37	0.80	0.69
Fine-tuned ESM2 models				
ESM2 150M frozen w. + linear layer + sigmoid	0.69	**0.38**	**0.85**	0.60
ESM2 150M + linear layer + sigmoid	0.71	0.35	0.79	0.67
Third-party tools				
BepiPred3 regions only	0.54	0.05	0.54	0.56
BepiPred3 full sequences	0.52	0.03	0.52	0.53
LBtope	0.53	0.04	0.52	0.55

modifications were tried. The AdamW optimizer was used for training with a reduction of the learning rate on a plateau. In the first experiment, the ESM2 weights were frozen, and only the additional layers were trained. Most of the models trained this way were unable to minimize training loss and resulted in MCC scores below 0.1. Multiple architectures were tested: using concatenation and difference of the wild-type and mutated embeddings, as well as single, double, and triple layer feed-forward neural networks. Furthermore, multiple different initial learning rates were tested: 0.1, 0.01, 0.001, 0.0001. None of these modifications made any difference in the MCC score, which was close to 0. Even if the multilayer networks with concatenation strategy were able to minimize the training loss, the validation loss increased, suggesting overfitting. Only after replacing the average pooling with [CLS] and [SEP] token embeddings, the models were able to achieve a higher validation MCC of around 0.3 but they were unable to optimize training loss, further proving that these tokens contain valuable information about the input sequence. Adding more layers to the neural network did not change the performance in any way for the [CLS] and [SEP] token embeddings. In general, taking the difference of the embeddings turned out to be a much worse strategy than concatenating the wild-type and mutant embeddings, yielding very low validation MCC scores (around 0.01) even for [CLS] token embeddings.

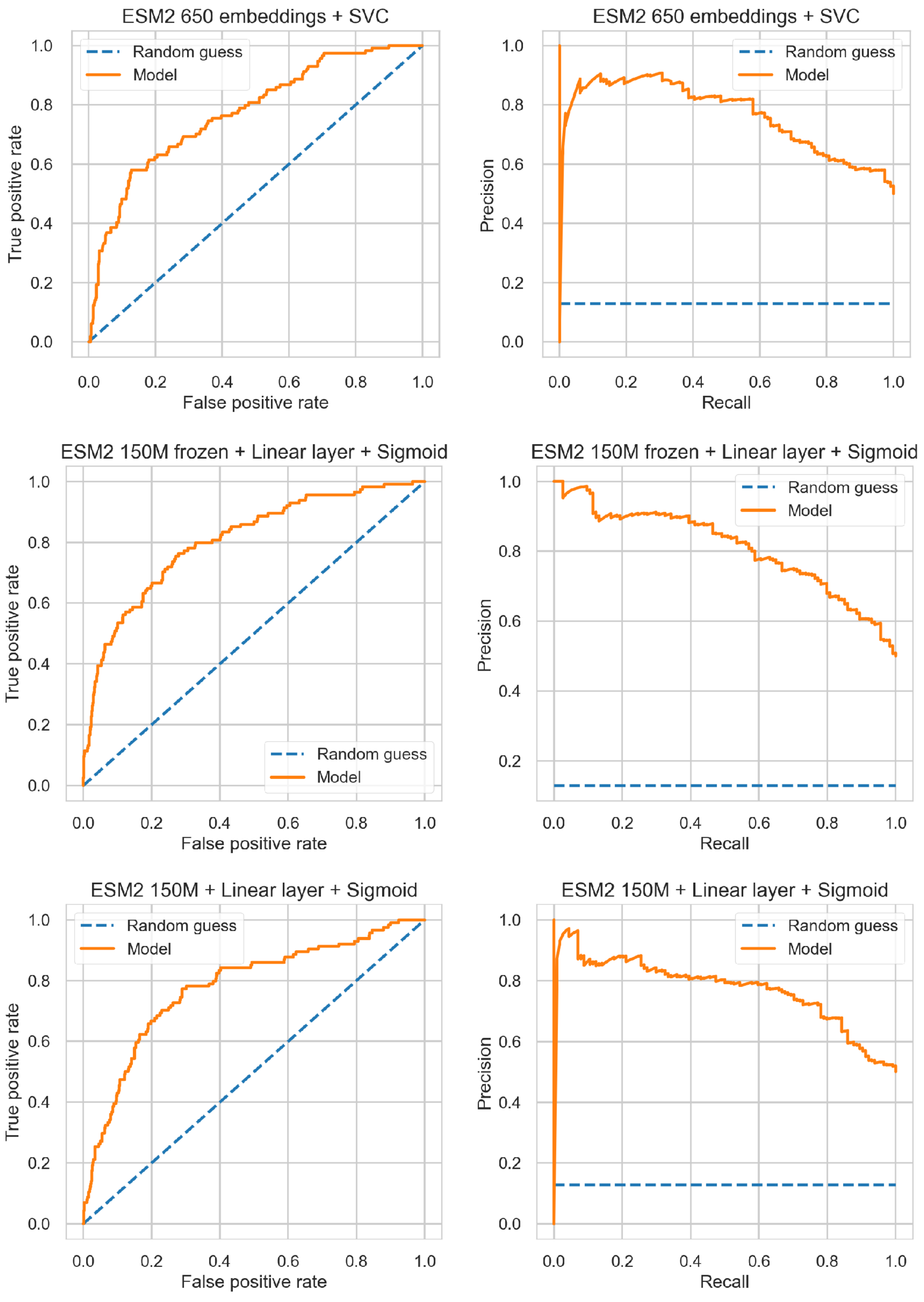
Finally, all the ESM2 weights were also trained. These models were able to optimize training loss much better, and most of them converged after 10 to 20 epochs. However, the validation MCC score did not improve and stayed at 0.3 on average. Furthermore, changing the other parameters did not change the overall performance, sometimes even worsening it. In addition, a trainable pooling layer was tested for aggregation, but the MCC score dropped to 0.26 on average. Again, the best-performing network used **CLS token** as the aggregation method and **concatenation of wild-type and mutated sequences**. It achieved an MCC score of 0.38 and balanced accuracy of 0.69. Unfortunately, for all models, the validation loss did not drop below 0.5. The best model variant was evaluated on the test set with a balanced accuracy score of 0.61 and an MCC score of 0.32.

The Precision-recall and ROC curves were calculated for the best models from the three rounds, and the graphs can be seen in Figure 7.2. Even though the ESM embeddings with SVC model had the best MCC score, the ROC curve suggests that the ESM2 with frozen weights, linear layer, and sigmoid should have a slightly higher true positive rate and lower false positive rate with the appropriate cutoff. The Precision-recall curves suggest that if we want to have precision above 80%, then the recall will be around 0.6, which could already help in reducing the potential promising mutations for experimental validation.

The hyperparameters of the best-performing model were used to train the model again, but on the *random split dataset*, to better understand the effect of the elaborate data-splitting strategy. Immediately, after the second epoch, the model achieved an MCC of 0.53 on the validation set and a balanced accuracy of 0.78. Thus, proving that robust dataset separation is essential when creating datasets and evaluating models with data from the same source on which they were trained.

7.3 Third-party tools

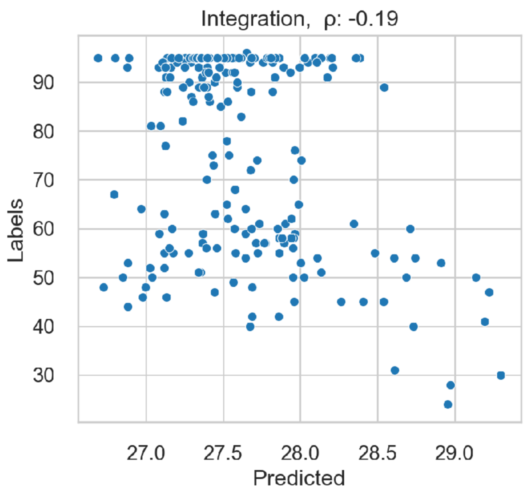
Since none of the third-party models is focused on the effects of mutations, their outputs were adjusted for this task. This resulted in poor scores for both models with accuracy close to 0.5 and MCC scores very close to 0, which is close to random guessing. The scores for both tools can be seen in Table 7.1. The BepiPred tool was evaluated using two strategies: when it is run only on regions and on the entire protein sequences.



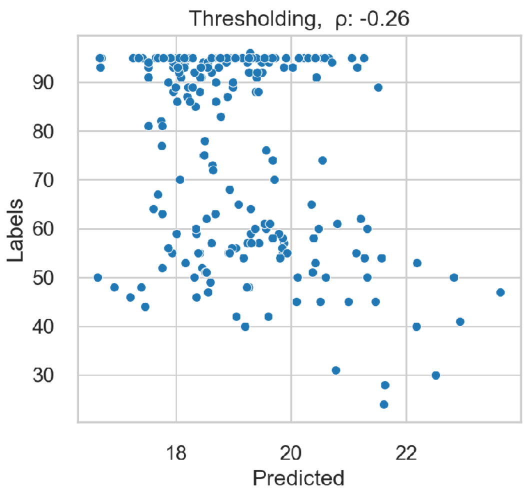
(a) ROC curves (b) Precision-Recall curves

Figure 7.2: ROC and Precision-Recall curves for selected models indicated in the subgraph titles.

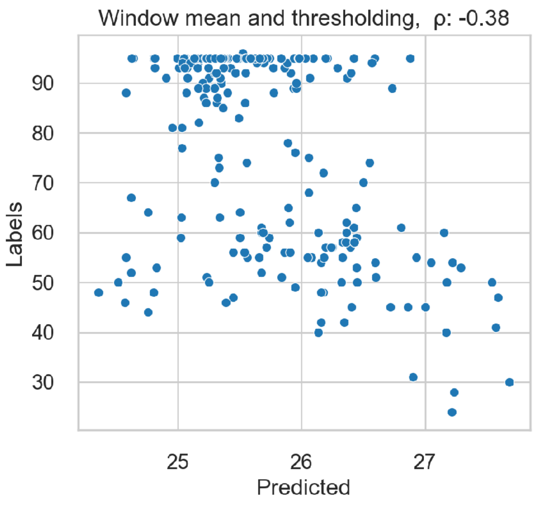
We also evaluated the BepiPred3 tool on the Staphylokinase dataset by calculating the average score over the whole sequence and comparing it with the experimental data from the study (see Figure 7.3. Since there are no known B-cell epitopes in IEDB for the Staphylokinase, there is no way to make the evaluation specific to certain regions. That is the reason the evaluation was done on the whole sequence, and why it is not possible to evaluate our predictors using this dataset.



(a)



(b)



(c)

Figure 7.3: Correlation of BepiPred3 predictions on the Staphylokinase dataset vs. real labels. The three graphs illustrate the results of different aggregation strategies of per-residue scores: (a) Summarization of the scores; (b) Summarization of scores that are above threshold of 0.2; (c) The scores are averaged using a moving window of 9 and then filtered using threshold. None of the strategies provide satisfactory result that would correlate with the real labels.

Chapter 8

Conclusion

Predicting the impact of mutations on antigen immunogenicity is an emerging field with significant implications for drug design. Currently, no specialized tools exist to address this specific challenge. While existing tools designed for epitope prediction offer some value, they fall short in accurately predicting the effects of mutations.

To address this gap, we created a novel dataset from the Immune Epitope Database (IEDB). Our approach involved identifying highly similar epitopes with demonstrably different immunogenicity profiles. By analyzing the sequence variations between these epitopes, we were able to capture the mutational effects on immunogenicity. This newly constructed dataset comprises 8,742 data points, with 1,584 entries specifically classified as exhibiting reduced immunogenicity.

Furthermore, we implemented two rigorous methods for creating training-test splits to ensure the absence of data leakage, a critical step in maintaining the generalizability of our models.

We then explored the efficacy of various machine-learning techniques for this novel task. This included employing linear models, such as Logistic Regression and Support Vector Classifiers, alongside large language models, specifically the ESM2 model. Interestingly, the most effective approach involved a hybrid strategy, leveraging the feature generation capabilities of ESM2 models to provide input for a Support Vector Classifier trained to predict the impact on immunogenicity. This combined model achieved an MCC score of 0.37 and a balanced accuracy of 0.73 on the held-out test set.

The newly created dataset presents a valuable foundation for the development of a user-friendly predictor tool. Such a tool could empower researchers to strategically introduce mutations into epitopes and lower immunogenicity while preserving the desired therapeutic function. Future endeavors to enhance model performance could involve further data cleaning based on additional criteria, such as assay type, or incorporating weights based on the number of assays supporting the immunogenicity classification.

In conclusion, this work establishes a crucial first step toward developing a new generation of tools for predicting the effects of mutations on immunogenicity. The novel dataset and the promising initial results pave the way for further advancements in this field, ultimately contributing to the design of more effective and targeted therapeutics.

Bibliography

[1] Abbas, A. K., Lichtman, A. H. and Pillai, S. *Basic Immunology E-Book*. Walthm, MA, USA: Elsevier, january 2019.

[2] Alberts, B., Johnson, A., Lewis, J., Raff, M., Roberts, K. et al. *Molecular Biology of the Cell*. New York, NY, USA: Garland Science, 2002. ISBN 978-0-8153-3218. Available at: https://www.ncbi.nlm.nih.gov/books/NBK21054.

[3] Alt, F. W., Oltz, E. M., Young, F., Gorman, J., Taccioli, G. et al. VDJ recombination. *Immunol. Today*. Elsevier. january 1992, vol. 13, no. 8, p. 306–314. DOI: 10.1016/0167-5699(92)90043-7. ISSN 0167-5699.

[4] Alwosheel, A., Cranenburgh, S. van and Chorus, C. G. Is your dataset big enough? Sample size requirements when using artificial neural networks for discrete choice analysis. *Journal of Choice Modelling*. Elsevier. september 2018, vol. 28, p. 167–182. DOI: 10.1016/j.jocm.2018.07.002. ISSN 1755-5345.

[5] Chen, Y., Lu, H., Zhang, N., Zhu, Z., Wang, S. et al. PremPS: Predicting the impact of missense mutations on protein stability. *PLoS Comput. Biol.* Public Library of Science. december 2020, vol. 16, no. 12, p. e1008543. DOI: 10.1371/journal.pcbi.1008543. ISSN 1553-7358.

[6] Cho, K., Merrienboer, B. van, Gulcehre, C., Bahdanau, D., Bougares, F. et al. Learning Phrase Representations using RNN Encoder-Decoder for Statistical Machine Translation. *ArXiv*. june 2014. DOI: 10.48550/arXiv.1406.1078.

[7] Clifford, J. N., Høie, M. H., Deleuran, S., Peters, B., Nielsen, M. et al. BepiPred-3.0: Improved B-cell epitope prediction using protein language models. *Protein Sci.* 2022 The Authors. Protein ScienceWiley Periodicals LLC on behalf of The Protein Society. december 2022, vol. 31, no. 12, p. e4497. DOI: 10.1002/pro.4497. ISSN 1469-896X.

[8] Clifford, J. N., Høie, M. H., Deleuran, S., Peters, B., Nielsen, M. et al. BepiPred-3.0: Improved B-cell epitope prediction using protein language models. *Protein Sci.* 2022 The Authors. Protein ScienceWiley Periodicals LLC on behalf of The Protein Society. december 2022, vol. 31, no. 12, p. e4497. DOI: 10.1002/pro.4497. ISSN 1469-896X.

[9] Cohan, A., Beltagy, I., King, D., Dalvi, B. and Weld, D. S. Pretrained Language Models for Sequential Sentence Classification. *ArXiv*. september 2019. DOI: 10.18653/v1/D19-1383.

[10] COLLEN, D. Engineered staphylokinase variants with reduced immunogenicity. *Fibrinolysis and Proteolysis*. Churchill Livingstone. september 1998, vol. 12, p. 59–65. DOI: 10.1016/S0268-9499(98)80307-X. ISSN 1369-0191.

[11] COLLEN, D. and LIJNEN, R. H. Thrombolytic agents. *Thromb. Haemost.* Schattauer GmbH. 2005, vol. 93, no. 04, p. 627–630. DOI: 10.1160/TH04-11-0724. ISSN 0340-6245.

[12] DE GROOT, A. S. and SCOTT, D. W. Immunogenicity of protein therapeutics. *Trends Immunol.* Elsevier. november 2007, vol. 28, no. 11, p. 482–490. DOI: 10.1016/j.it.2007.07.011. ISSN 1471-4906.

[13] DEVLIN, J., CHANG, M.-W., LEE, K. and TOUTANOVA, K. BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding. *ArXiv.* october 2018. DOI: 10.48550/arXiv.1810.04805.

[14] DU, Z., DING, X., HSU, W., MUNIR, A., XU, Y. et al. pLM4ACE: A protein language model based predictor for antihypertensive peptide screening. *Food Chem.* Elsevier. january 2024, vol. 431, p. 137162. DOI: 10.1016/j.foodchem.2023.137162. ISSN 0308-8146.

[15] EINARSON, T. R., BOOTMAN, J. L. and SMITH, G. H. Chymopapain. *Drug Intell. Clin. Pharm.* SAGE Publications. july 1984, vol. 18, 7-8, p. 560–568. DOI: 10.1177/106002808401800702. ISSN 0012-6578.

[16] ELNAGGAR, A., HEINZINGER, M., DALLAGO, C., REHAWI, G., WANG, Y. et al. ProtTrans: Toward Understanding the Language of Life Through Self-Supervised Learning. *IEEE Trans. Pattern Anal. Mach. Intell.* IEEE. july 2021, vol. 44, no. 10, p. 7112–7127. DOI: 10.1109/TPAMI.2021.3095381.

[17] FARHANA, A. and LAPPIN, S. L. Biochemistry, Lactate Dehydrogenase. In: *StatPearls [Internet].* StatPearls Publishing, May 2023. Available at: https://www.ncbi.nlm.nih.gov/books/NBK557536.

[18] FERDOUS, S. and MARTIN, A. C. R. AbDb: antibody structure database—a database of PDB-derived antibody structures. *Database.* Oxford Academic. january 2018, vol. 2018, p. bay040. DOI: 10.1093/database/bay040. ISSN 1758-0463.

[19] GELMAN, S., FAHLBERG, S. A., HEINZELMAN, P., ROMERO, P. A. and GITTER, A. Neural networks to learn protein sequence–function relationships from deep mutational scanning data. *Proceedings of the National Academy of Sciences.* National Acad Sciences. 2021, vol. 118, no. 48, p. e2104878118.

[20] GREENSPAN, N. S. and DI CERA, E. Defining epitopes: It's not as easy as it seems. *Nat. Biotechnol.* Nature Publishing Group. october 1999, vol. 17, no. 10, p. 936–937. DOI: 10.1038/13590. ISSN 1546-1696.

[21] GUNTUBOINA, C., DAS, A., MOLLAEI, P., KIM, S. and FARIMANI, A. B. PeptideBERT: A Language Model based on Transformers for Peptide Property Prediction. *ArXiv.* august 2023. DOI: 10.48550/arXiv.2309.03099.

[22] Gusev, E. I. and Martynov, M. Y. c. Non-immunogenic recombinant staphylokinase versus alteplase for patients with acute ischaemic stroke 4·5 h after symptom onset in Russia (FRIDA): a randomised, open label, multicentre, parallel-group, non-inferiority trial. *Lancet Neurol.* Elsevier. september 2021, vol. 20, no. 9, p. 721–728. DOI: 10.1016/S1474-4422(21)00210-6. ISSN 1474-4422.

[23] Hede, P. D. *A beginner's guide to enzymes in detergents*. January 2024. [Online; accessed 10. Jan. 2024]. Available at: https://biosolutions.novozymes.com/en/dish/insights/article/beginners-guide-enzymes-detergents.

[24] Hochreiter, S. The Vanishing Gradient Problem During Learning Recurrent Neural Nets and Problem Solutions. *Int. J. Uncertainty Fuzziness Knowledge-Based Syst.* World Scientific Publishing Co. april 1998, vol. 06, no. 02, p. 107–116. DOI: 10.1142/S0218488598000094. ISSN 0218-4885.

[25] Hochreiter, S. and Schmidhuber, J. Long Short-term Memory. *Neural Comput.* MIT Press. december 1997, vol. 9, no. 8, p. 1735–80. DOI: 10.1162/neco.1997.9.8.1735. ISSN 0899-7667.

[26] Ishack, S. and Lipner, S. R. Bioinformatics and immunoinformatics to support COVID-19 vaccine development. *J. Med. Virol.* Wiley-Blackwell. september 2021, vol. 93, no. 9, p. 5209. DOI: 10.1002/jmv.27017.

[27] Jespersen, M. C., Peters, B., Nielsen, M. and Marcatili, P. BepiPred-2.0: improving sequence-based B-cell epitope prediction using conformational epitopes. *Nucleic Acids Res.* The Author(s) 2017Oxford University Press on behalf of Nucleic Acids Research. july 2017, vol. 45, W1, p. 24–29. DOI: 10.1093/nar/gkx346. ISSN 1362-4962.

[28] Jumper, J., Evans, R., Pritzel, A., Green, T., Figurnov, M. et al. Highly accurate protein structure prediction with AlphaFold. *Nature.* Nature Publishing Group. august 2021, vol. 596, no. 7873, p. 583–589. DOI: 10.1038/s41586-021-03819-2. ISSN 1476-4687.

[29] Kaabinejadian, S., Barra, C., Alvarez, B., Yari, H., Hildebrand, W. H. et al. Accurate MHC Motif Deconvolution of Immunopeptidomics Data Reveals a Significant Contribution of DRB3, 4 and 5 to the Total DR Immunopeptidome. *Front. Immunol.* Kaabinejadian, Barra, Alvarez, Yari, Hildebrand and Nielsen. january 2022, 13:835454. DOI: 10.3389/fimmu.2022.835454. ISSN 1664-3224.

[30] Katoh, K. and Standley, D. M. MAFFT Multiple Sequence Alignment Software Version 7: Improvements in Performance and Usability. *Mol. Biol. Evol.* Oxford University Press. april 2013, vol. 30, no. 4, p. 772. DOI: 10.1093/molbev/mst010.

[31] Kew, B., Holmes, M., Liamas, E., Ettelaie, R., Connell, S. D. et al. Transforming sustainable plant proteins into high performance lubricating microgels. *Nat. Commun.* Nature Publishing Group. august 2023, vol. 14, no. 4743, p. 1–15. DOI: 10.1038/s41467-023-40414-7. ISSN 2041-1723.

[32] Khurana, D., Koli, A., Khatter, K. and Singh, S. Natural language processing: state of the art, current trends and challenges. *Multimed. Tools Appl.* Springer US.

january 2023, vol. 82, no. 3, p. 3713–3744. DOI: 10.1007/s11042-022-13428-4. ISSN 1573-7721.

[33] Knox, C., Wilson, M., Klinger, C. M., Franklin, M., Oler, E. et al. DrugBank 6.0: the DrugBank Knowledgebase for 2024. *Nucleic Acids Res.* The Author(s) 2023Oxford University Press on behalf of Nucleic Acids Research. january 2024, vol. 52, D1, p. 1265–1275. DOI: 10.1093/nar/gkad976. ISSN 1362-4962.

[34] Kouba, P., Kohout, P., Haddadi, F., Bushuiev, A., Samusevich, R. et al. Machine Learning-Guided Protein Engineering. *ACS Catal.* American Chemical Society. november 2023, vol. 13, no. 21, p. 13863–13895. DOI: 10.1021/acscatal.3c02743.

[35] Kulmanov, M., Guzmán Vega, F. J., Duek Roggli, P., Lane, L., Arold, S. T. et al. Protein function prediction as approximate semantic entailment. *Nat. Mach. Intell.* Nature Publishing Group. february 2024, vol. 6, no. 2, p. 220–228. DOI: 10.1038/s42256-024-00795-w. ISSN 2522-5839.

[36] Lin, Z., Akin, H., Rao, R., Hie, B., Zhu, Z. et al. Evolutionary-scale prediction of atomic-level protein structure with a language model. *Science.* American Association for the Advancement of Science. march 2023, vol. 379, no. 6637, p. 1123–1130. DOI: 10.1126/science.ade2574. ISSN 0036-8075.

[37] Liu, X., Duh, K., Liu, L. and Gao, J. Very Deep Transformers for Neural Machine Translation. *ArXiv.* august 2020. DOI: 10.48550/arXiv.2008.07772.

[38] Lu, H., Diaz, D. J., Czarnecki, N. J., Zhu, C., Kim, W. et al. Machine learning-aided engineering of hydrolases for PET depolymerization. *Nature.* Nature Publishing Group. april 2022, vol. 604, no. 7907, p. 662–667. DOI: 10.1038/s41586-022-04599-z. ISSN 1476-4687.

[39] Madani, M., Lin, K. and Tarakanova, A. DSResSol: A Sequence-Based Solubility Predictor Created with Dilated Squeeze Excitation Residual Networks. *Int. J. Mol. Sci.* Int J Mol Sci. december 2021, vol. 22, no. 24, p. 13555. DOI: 10.3390/ijms222413555. ISSN 1422-0067.

[40] Marrack, P. and Kappler, J. The T Cell Receptor. *Science.* American Association for the Advancement of Science. november 1987, vol. 238, no. 4830, p. 1073–1079. DOI: 10.1126/science.3317824. ISSN 0036-8075.

[41] Matthews, B. W. Comparison of the predicted and observed secondary structure of T4 phage lysozyme. *Biochimica et Biophysica Acta (BBA) - Protein Structure.* Elsevier. october 1975, vol. 405, no. 2, p. 442–451. DOI: 10.1016/0005-2795(75)90109-9. ISSN 0005-2795.

[42] Mitchell, T. *Machine Learning.* McGraw-Hill, 1997. McGraw-Hill International Editions. ISBN 9780071154673. Available at: https://books.google.cz/books?id=EoYBngEACAAJ.

[43] Müller, A. and Guido, S. *Introduction to Machine Learning with Python: A Guide for Data Scientists.* O'Reilly Media, 2016. ISBN 9781449369897. Available at: https://books.google.cz/books?id=vbQlDQAAQBAJ.

[44] Musil, M., Jezik, A., Horackova, J., Borko, S., Kabourek, P. et al. FireProt 2.0: web-based platform for the fully automated design of thermostable proteins. *Briefings Bioinf.* The Author(s) 2023Oxford University Press. november 2023, vol. 25, no. 1, p. bbad425. DOI: 10.1093/bib/bbad425. ISSN 1477-4054.

[45] Noelle, R. J. and Snow, E. C. Cognate interactions between helper T cells and B cells. *Immunol. Today.* Elsevier. january 1990, vol. 11, p. 361–368. DOI: 10.1016/0167-5699(90)90142-V. ISSN 0167-5699.

[46] Ong, E., Cooke, M. F., Huffman, A., Xiang, Z., Wong, M. U. et al. Vaxign2: the second generation of the first Web-based vaccine design program using reverse vaccinology and machine learning. *Nucleic Acids Res.* Oxford University Press. july 2021, vol. 49, W1, p. W671. DOI: 10.1093/nar/gkab279.

[47] Ong, E., Cooke, M. F., Huffman, A., Xiang, Z., Wong, M. U. et al. Vaxign2: the second generation of the first Web-based vaccine design program using reverse vaccinology and machine learning. *Nucleic Acids Res.* Oxford University Press. july 2021, vol. 49, W1, p. W671. DOI: 10.1093/nar/gkab279.

[48] Ong, E., Wong, M. U., Huffman, A. and He, Y. COVID-19 Coronavirus Vaccine Design Using Reverse Vaccinology and Machine Learning. *Front. Immunol.* Frontiers Media SA. 2020, vol. 11. DOI: 10.3389/fimmu.2020.01581.

[49] OpenAI, Achiam, J., Adler, S. and co.. GPT-4 Technical Report. *ArXiv.* march 2023. DOI: 10.48550/arXiv.2303.08774.

[50] Paul, S., Sidney, J., Sette, A. and Peters, B. TepiTool: A Pipeline for Computational Prediction of T Cell Epitope Candidates. *Current Protocols in Immunology.* John Wiley & Sons, Ltd. august 2016, vol. 114, no. 1, p. 18.19.1–18.19.24. DOI: 10.1002/cpim.12. ISSN 1934-3671.

[51] Plotkin, S., Robinson, J. M., Cunningham, G., Iqbal, R. and Larsen, S. The complexity and cost of vaccine manufacturing – An overview. *Vaccine.* Elsevier. july 2017, vol. 35, no. 33, p. 4064. DOI: 10.1016/j.vaccine.2017.06.003.

[52] Ransbotham, S. and Johnson, D. *AI and the COVID-19 Vaccine: Moderna's Dave Johnson.* April 2024. [Online; accessed 11. Apr. 2024]. Available at: https://sloanreview.mit.edu/audio/ai-and-the-covid-19-vaccine-modernas-dave-johnson.

[53] Reynisson, B., Alvarez, B., Paul, S., Peters, B. and Nielsen, M. NetMHCpan-4.1 and NetMHCIIpan-4.0: improved predictions of MHC antigen presentation by concurrent motif deconvolution and integration of MS MHC eluted ligand data. *Nucleic Acids Res.* Oxford University Press. july 2020, vol. 48, W1, p. W449. DOI: 10.1093/nar/gkaa379.

[54] Reynisson, B., Alvarez, B., Paul, S., Peters, B. and Nielsen, M. NetMHCpan-4.1 and NetMHCIIpan-4.0: improved predictions of MHC antigen presentation by concurrent motif deconvolution and integration of MS MHC eluted ligand data. *Nucleic Acids Res.* Oxford University Press. july 2020, vol. 48, W1, p. W449. DOI: 10.1093/nar/gkaa379.

[55] Rodrigues, C. H. M., Pires, D. E. V. and Ascher, D. B. DynaMut2: Assessing changes in stability and flexibility upon single and multiple point missense mutations. *Protein Sci.* John Wiley & Sons, Ltd. january 2021, vol. 30, no. 1, p. 60–69. DOI: 10.1002/pro.3942. ISSN 0961-8368.

[56] Root, M. A., Chance, R. E. and Galloway, J. A. Immunogenicity of Insulin. *Diabetes.* American Diabetes Association. june 1972, vol. 21, Supplement_2, p. 657–660. DOI: 10.2337/diab.21.2.S657. ISSN 0012-1797.

[57] Samuel, A. L. Some Studies in Machine Learning Using the Game of Checkers. *IBM Journal of Research and Development.* 1959, vol. 3, no. 3, p. 210–229. DOI: 10.1147/rd.33.0210.

[58] Schak, M. and Gepperth, A. A Study on Catastrophic Forgetting in Deep LSTM Networks. In: *Artificial Neural Networks and Machine Learning – ICANN 2019: Deep Learning.* Cham, Switzerland: Springer, September 2019, p. 714–728. DOI: 10.1007/978-3-030-30484-3_56. ISBN 978-3-030-30484-3.

[59] Schmirler, R., Heinzinger, M. and Rost, B. Fine-tuning protein language models boosts predictions across diverse tasks. *BioRxiv.* Cold Spring Harbor Laboratory. december 2023, p. 2023.12.13.571462. DOI: 10.1101/2023.12.13.571462.

[60] Silva, B. M. da, Ascher, D. B. and Pires, D. E. V. epitope1D: accurate taxonomy-aware B-cell linear epitope prediction. *Briefings Bioinf.* Oxford University Press. may 2023, vol. 24, no. 3. DOI: 10.1093/bib/bbad114.

[61] Singh, H., Ansari, H. R. and Raghava, G. P. S. Improved Method for Linear B-Cell Epitope Prediction Using Antigen's Primary Sequence. *PLoS One.* Public Library of Science. may 2013, vol. 8, no. 5, p. e62216. DOI: 10.1371/journal.pone.0062216. ISSN 1932-6203.

[62] Steinegger, M. and Söding, J. MMseqs2 enables sensitive protein sequence searching for the analysis of massive data sets. *Nat. Biotechnol.* Nature Publishing Group. november 2017, vol. 35, no. 11, p. 1026–1028. DOI: 10.1038/nbt.3988. ISSN 1546-1696.

[63] Team, G., Anil, R., Borgeaud, S. and co.. Gemini: A Family of Highly Capable Multimodal Models. *ArXiv.* december 2023. DOI: 10.48550/arXiv.2312.11805.

[64] Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L. et al. Attention Is All You Need. *ArXiv.* june 2017. DOI: 10.48550/arXiv.1706.03762.

[65] Verkuil, R., Kabeli, O., Du, Y., Wicky, B. I. M., Milles, L. F. et al. Language models generalize beyond natural proteins. *BioRxiv.* Cold Spring Harbor Laboratory. december 2022, p. 2022.12.21.521521. Available at: https://doi.org/10.1101/2022.12.21.521521.

[66] Vita, R., Mahajan, S., Overton, J. A., Dhanda, S. K., Martini, S. et al. The Immune Epitope Database (IEDB): 2018 update. *Nucleic Acids Res.* See full text options at Silverchair Information Systems. january 2019, vol. 47, D1, p. 339–343. DOI: 10.1093/nar/gky1006. ISSN 1362-4962.

[67] Watson, J. L., Juergens, D., Bennett, N. R., Trippe, B. L., Yim, J. et al. De novo design of protein structure and function with RFdiffusion. *Nature*. Nature Publishing Group. august 2023, vol. 620, no. 7976, p. 1089–1100. DOI: 10.1038/s41586-023-06415-8. ISSN 1476-4687.

[68] Wu, R., Ding, F., Wang, R., Shen, R., Zhang, X. et al. High-resolution de novo structure prediction from primary sequence. *BioRxiv*. Cold Spring Harbor Laboratory. 2022. DOI: 10.1101/2022.07.21.500999. Available at: https://www.biorxiv.org/content/early/2022/07/22/2022.07.21.500999.

[69] Zinsli, L. V., Stierlin, N., Loessner, M. J. and Schmelcher, M. Deimmunization of protein therapeutics – Recent advances in experimental and computational epitope prediction and deletion. *Comput. Struct. Biotechnol. J.* Research Network of Computational and Structural Biotechnology. 2021, vol. 19, p. 315. DOI: 10.1016/j.csbj.2020.12.024.